GT2: DIRECTING WORLD MODELING VIA GEOMETRY -GUIDED TEST-TIME OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Video world models offer rich priors for simulating dynamic environments. However, jointly controlling camera and object motions remains challenging because their effects are coupled in the observed video and must be composed consistently in a shared 3D coordinate system. Existing methods typically address this challenge through task-specific training on large-scale videos with detailed camera and object motion annotations, making joint control resource-intensive and difficult to adapt. We propose GT2 (Geometry-Guided Test-Time Optimization), a framework that enables joint camera and object control in pretrained video generators by providing explicit geometric supervision at test time, without additional large-scale training. Specifically, GT2 composes prescribed camera motions and object translations in a shared 3D coordinate system and projects the input scene to construct a coarse preview of their combined visual effects. This geometry-aware preview serves as supervision for test-time optimization, directly steering generation toward the prescribed scene dynamics. Experiments show that GT2 improves joint motion controllability while maintaining high visual quality, reducing camera translation error, camera rotation error, and object control error by approximately 28%, 6%, and 10%, respectively, compared with leading baselines.

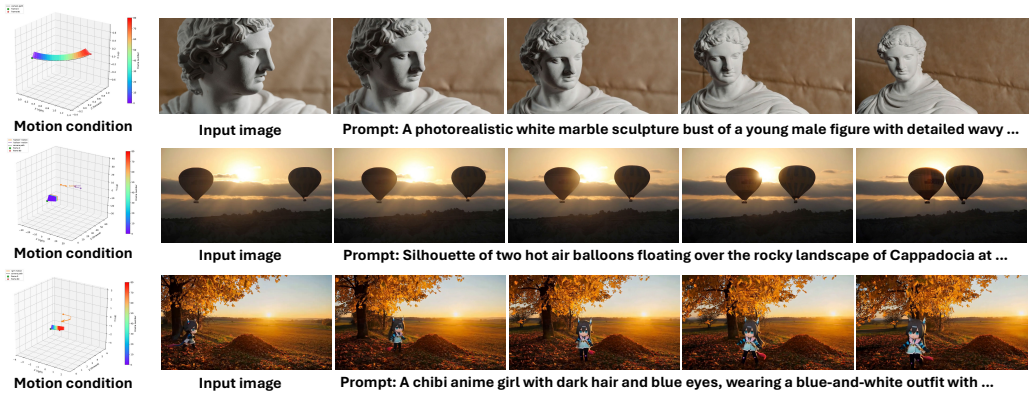

Figure 1: From top to bottom, we show camera control with a fixed object, object control with a fixed camera, and joint control with both camera and object motions prescribed. Across each row, GT2 follows the specified motion conditions while maintaining consistent scene appearance.

1 INTRODUCTION

World models predict the future dynamics of an environment from past observations and action signals (Ha & Schmidhuber, 2018; Koh et al., 2021; Bar et al., 2025). Recent high-fidelity video generators (Blattmann et al., 2023; Kong et al., 2024; Wan et al., 2025; Yang et al., 2025) provide strong visual priors for this goal, enabling an observed scene to be rolled forward into plausible future videos. Interactive world simulation and cinematic content creation, however, require more than plausible prediction: users need to specify how both the camera and objects should move. Jointly realizing these controls is challenging because camera and object motions produce coupled effects

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

after projection onto the image plane. The same apparent displacement can result from a viewpoint change, an object movement, or their combination. A controllable video world model must therefore generate the combined visual effects of camera and object motions composed consistently in a shared 3D coordinate system.

Existing methods provide only partial solutions to this problem. Camera-oriented approaches (He et al., 2025a; Bahmani et al., 2025; Song et al., 2026) primarily control viewpoint changes while leaving object motion static or unconstrained, whereas object-centric approaches (Shi et al., 2024; Li et al., 2025a; Fu et al., 2025) manipulate objects under a fixed or uncontrolled viewpoint. Neither setting fully addresses the need to coordinate both sources of motion. Recent joint-control methods (Chen et al., 2025b; Zhang et al., 2026; Zheng et al., 2026b) address this requirement by decoupling camera and object controls into separate modules or integrating them through a unified control adapter. However, they typically acquire this capability through task-specific training on large-scale videos with detailed camera trajectories and object motion annotations in a shared 3D coordinate system. Achieving joint control through this route therefore entails substantial data preparation and offline training costs.

We formulate joint motion control as geometry-guided test-time optimization. Given an initial image, a prescribed camera trajectory, and desired 3D translations for masked objects, our objective is to generate a video that follows the specified scene dynamics, as illustrated in Fig. 1. Pretrained video generators provide rich appearance and motion priors but lack a native interface for these 3D controls. Our key idea is to use geometry to translate the controls into visual supervision: we explicitly compose the prescribed motions, project their combined effects onto the image plane, and use the resulting preview to guide generation at test time. This enables joint control without learning a task-specific controller from annotated videos, trading additional per-video optimization for avoiding large-scale offline training.

Specifically, we propose GT2, a two-stage framework consisting of geometry-aware previsualization and test-time optimization. In the previsualization stage, we estimate the input scene's depth and project the scene under the prescribed camera transformations and object translations to obtain a coarse preview of their combined effects. This projection can expose previously unobserved regions and introduce warping artifacts, making the preview unsuitable for uniform supervision. We therefore use depth-aware coarse inpainting to complete missing areas and combine background erosion with projected object Gaussian cores to retain reliable supervision regions. For the objects, we adopt the Gaussian representation from VerseCrafter (Zheng et al., 2026b). The resulting preview serves as a geometric scaffold that specifies where reliable motion evidence should appear. In the test-time optimization stage, we apply localized supervision from this scaffold during early denoising steps, updating lightweight LoRA (Hu et al., 2021) parameters while keeping the video generator's backbone frozen. The geometric supervision guides motion, while the pretrained video prior supplies appearance and fine details.

We evaluate GT2 on a curated benchmark of 100 joint-control samples. Extensive experiments demonstrate improved motion controllability while maintaining high visual quality. Compared with leading baselines, GT2 reduces camera translation error, camera rotation error, and object-control error by approximately 28%, 6%, and 10%, respectively, without large-scale task-specific data preparation or offline controller training.

2 METHODOLOGY

In this section, we introduce GT2, a two-stage framework for joint camera and object control. As shown in Fig. 2, the previsualization stage (Sec. 2.1) first converts the prescribed camera trajectory and object translations into a coarse, geometry-aware preview. The test-time optimization stage (Sec. 2.2) then uses reliable regions of this preview to guide early denoising steps, steering the generated video toward the prescribed motion conditions. We use s to index video frames and t to index diffusion timesteps throughout this section.

2.1 PREVISUALIZATION STAGE

The previsualization stage constructs a coarse, geometry-aware preview of the target scene dynamics before inference. Given a reference image I_0, we estimate metric depth D_0 and camera intrinsics

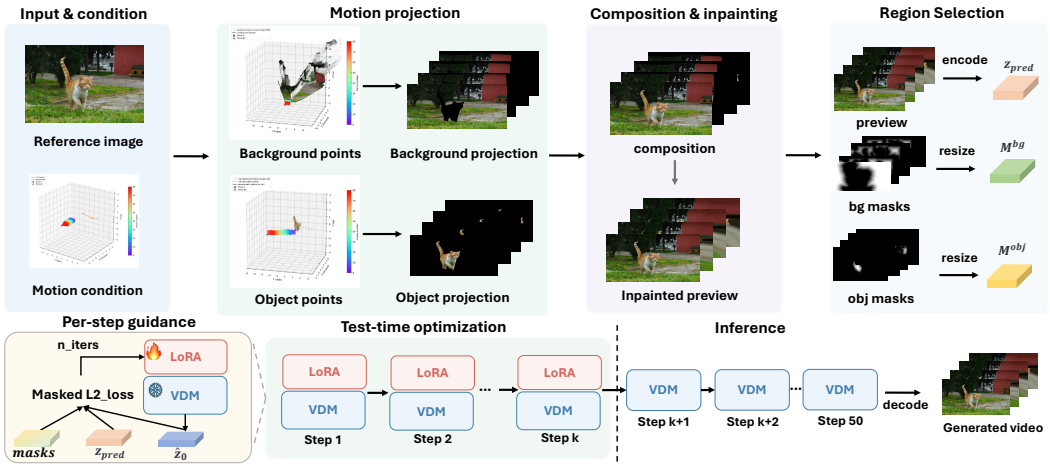

Figure 2: Overview of GT2. Given the reference image and motion conditions, we first construct an inpainted preview by projecting the prescribed camera–object motions from 3D world space to the image plane. During test-time optimization, LoRA parameters are optimized to align intermediate clean predictions from video diffusion models (VDMs) with reliable preview regions.

$\mathbf{K}$ using MoGe-v2 (Wang et al., 2025c), and extract O object segmentation masks $\{S_o\}_{o=1}^{O}$ with Grounded-SAM2 (Ren et al., 2024). For each pixel $p = (u, v, 1)^\top$, we back-project it to world space as $D_0(u, v)\mathbf{K}^{-1}p$ and attach the RGB value $I_0(u, v)$ to the resulting 3D point. We define the background as the complement of the union of all object masks, partitioning the reconstructed point cloud into background points $\mathbf{P}^{\text{bg}}$ and per-object point sets $\{\mathbf{P}^o\}_{o=1}^{O}$. Given a target camera trajectory represented by world-to-camera transforms $\{(\mathbf{R}_s, \mathbf{t}_s)\}_{s=0}^{S-1}$ with $\mathbf{R}_0 = \mathbf{I}, \mathbf{t}_0 = \mathbf{0}$, and per-object translations $\{\delta_o(s)\}_{s=0}^{S-1}$ with $\delta_o(0) = \mathbf{0}$, we warp and project the point sets along the prescribed trajectories to obtain the preview. Both camera and object translations are specified in meters in the MoGe coordinate system.

Background Projection. Given the static background point cloud $\mathbf{P}^{\text{bg}}$ reconstructed from the reference view, we project each point $\mathbf{p} \in \mathbf{P}^{\text{bg}}$ into the s-th camera frame as

$$\mathbf{u}_s(\mathbf{p}) = \pi\left(\mathbf{K}\left(\mathbf{R}_s\mathbf{p} + \mathbf{t}_s\right)\right), \tag{1}$$

where $\pi(\cdot)$ denotes perspective division. We discard points that project outside the image or fall behind the camera. The camera-space depth of each valid projected point is

$$z_s(\mathbf{p}) = \left[\mathbf{R}_s\mathbf{p} + \mathbf{t}_s\right]_z. \tag{2}$$

Object Projection. Each controllable object o is represented by a point set $\mathbf{P}^o$ extracted using its segmentation mask S_o. Let $\delta_o(s)$ denote its prescribed translation in the world coordinate system at frame s. We express the combined object and camera motion as an equivalent relative-camera transform:

$$\mathbf{R}_{s,o}^{\text{rel}}\mathbf{x} + \mathbf{t}_{s,o}^{\text{rel}} = \mathbf{R}_s\big(\mathbf{x} + \delta_o(s)\big) + \mathbf{t}_s, \tag{3}$$

where $\mathbf{x} \in \mathbf{P}^o$. This gives

$$\mathbf{R}_{s,o}^{\text{rel}} = \mathbf{R}_s, \quad \mathbf{t}_{s,o}^{\text{rel}} = \mathbf{t}_s + \mathbf{R}_s\delta_o(s). \tag{4}$$

We then project $\mathbf{P}^o$ using $(\mathbf{R}_{s,o}^{\text{rel}}, \mathbf{t}_{s,o}^{\text{rel}})$ and the same intrinsics $\mathbf{K}$.

The projected background and object points are jointly rasterized using a shared z-buffer, retaining at each pixel the candidate with the smallest camera-space depth. This unified depth test resolves inter-component occlusions and determines which pixels remain visible for the background and each object. We denote the resulting background and foreground projections after the depth test as $\text{Proj}_s^{\text{bg}}$ and $\text{Proj}_s^{\text{obj}}$, their depth maps as D_s^{bg} and D_s^{obj}, and the corresponding visibility masks as V_s^{bg} and $\{V_s^o\}_{o=1}^{O}$, respectively. This formulation places camera and object motions within a unified projection model while keeping all prescribed motions in a consistent world coordinate system.

Depth-aware Coarse Inpainting. Projection can expose regions that are not visible in the reference view, producing holes in the projected frames. Although these holes are excluded from the

supervision loss, leaving them empty can contaminate nearby latent features through the VAE's receptive field. We therefore coarsely fill them before VAE encoding. Let V_s^{bg} and V_s^o denote the pixels covered by the projected background and object o at frame s. A pixel $\mathbf{u}$ is defined as a hole if it lies outside all projected coverage, and we denote the hole set as

$$\Omega_s = \left\{ \mathbf{u} \mid \mathbf{u} \notin V_s^{\text{bg}},\ \mathbf{u} \notin \bigcup_o V_s^o \right\}. \tag{5}$$

To inpaint Ω_s, we first remove small cracks within the background and object masks using morphological closing and local neighbor averaging. For the remaining large holes, we perform depth-aware boundary propagation from valid background pixels. Specifically, let $\mathcal{N}(\mathbf{u})$ denote the local neighborhood of $\mathbf{u}$, and let $C_s(\mathbf{v})$ and $d_s(\mathbf{v})$ denote the color and depth of a valid background pixel $\mathbf{v}$. At each iteration, a hole pixel $\mathbf{u}$ adjacent to the valid region estimates its depth as

$$\hat{d}_s(\mathbf{u}) = \text{mean}_{\mathbf{v}\in\mathcal{N}(\mathbf{u})} d_s(\mathbf{v}), \tag{6}$$

and selects neighbors with consistent depths,

$$\mathcal{N}_\tau(\mathbf{u}) = \left\{ \mathbf{v} \in \mathcal{N}(\mathbf{u}) : |d_s(\mathbf{v}) - \hat{d}_s(\mathbf{u})| \le \tau\, \hat{d}_s(\mathbf{u}) \right\}, \tag{7}$$

where $\tau > 0$ is a relative depth tolerance. The hole pixel is then assigned the average color of these depth-consistent neighbors:

$$\hat{C}_s(\mathbf{u}) = \frac{1}{|\mathcal{N}_\tau(\mathbf{u})|} \sum_{\mathbf{v}\in\mathcal{N}_\tau(\mathbf{u})} C_s(\mathbf{v}). \tag{8}$$

Newly filled pixels are treated as valid for subsequent propagation, with their depths set to the mean depth of $\mathcal{N}_\tau(\mathbf{u})$. This process progressively shrinks holes from their boundaries inward until further propagation becomes negligible. The remaining unfilled regions are then inpainted by a distance transform operation. The final composition is

$$I_s^{\text{comp}} = \text{Proj}_s^{\text{bg}} \odot V_s^{\text{bg}} + \text{Proj}_s^{\text{obj}} \odot \bigcup_o V_s^o + \hat{C}_s \odot \mathbf{1}_{\Omega_s}, \tag{9}$$

yielding an inpainted preview for the subsequent test-time optimization.

2.2 Test-time Optimization Stage

Given the preview sequence $X_{\text{pre}} = \{I_s^{\text{comp}}\}_{s=0}^{S-1}$ together with the projected background and object masks, we perform test-time optimization to steer the denoising trajectory toward the prescribed camera and object motions.

Supervision Region Selection. Using the entire projected preview as supervision can introduce unreliable signals near projection and inpainting boundaries. We therefore construct separate reliability filters for the background and object regions.

For the background, we erode the projected background mask to remove boundary pixels that are more susceptible to projection artifacts. Specifically, we define

$$M_s^{\text{bg}} = \mathcal{E}_{r_{s,\text{bg}}} V_s^{\text{bg}} \tag{10}$$

where $\mathcal{E}_r(\cdot)$ denotes morphological erosion, and the per-frame erosion radius $r_{s,\text{bg}}$ is determined adaptively from valid pixels' Euclidean distances to the mask boundary.

For each object o, following VerseCrafter (Zheng et al., 2026b), we obtain a 2D Gaussian region $G_{s,o}^{2D}$ by projecting its estimated 3D Gaussian geometry $\mathcal{G}_o(\mathbf{x}) = \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_o, \boldsymbol{\Sigma}_o)$ to the image plane using the same relative camera transformation $(\mathbf{R}_{s,o}^{\text{rel}}, \mathbf{t}_{s,o}^{\text{rel}})$ as for the object point cloud. We then use the high-density core of the projected Gaussian as the object supervision region:

$$M_{s,o}^{\text{obj}} = \left\{ \mathbf{u} \in V_s^o \;\middle|\; \frac{G_{s,o}^{2D}(\mathbf{u})}{\max_{\mathbf{v}} G_{s,o}^{2D}(\mathbf{v}) + \epsilon} > \tau_{\text{obj}} \right\}, \tag{11}$$

where τ_{obj} controls the size of the Gaussian core, and the normalization is performed with respect to the maximum Gaussian response over the entire image.

This selection process retains the interior of the projected background and objects while suppressing unreliable boundary regions. We then encode X_{pre} with the video VAE to obtain z_{pre}, and align the corresponding masks with the VAE latent sequence by preserving the initial frame and max-pooling subsequent temporal groups, followed by spatial resizing to the latent resolution.

Test-time Optimization. Let v_θ denote the video backbone, whose pretrained parameters θ remain frozen. We introduce lightweight trainable LoRA parameters ϕ, shared across the optimized denoising steps, to inject the target motion constraints without updating the backbone.

Let z_t denote the current latent at diffusion step t. During the optimized steps, the LoRA-adapted backbone predicts the denoising velocity

$$\mathbf{v}_{\theta,\phi}(z_t, t) = v_{\theta,\phi}(z_t, t, c_\tau, c_I), \tag{12}$$

where c_τ and c_I denote the text and image conditions, respectively. Following the standard flow-matching formulation, the corresponding clean latent is estimated as

$$\hat{z}_0 = z_t - t\,\mathbf{v}_{\theta,\phi}(z_t, t). \tag{13}$$

We define separate reconstruction losses over the selected background and object regions:

$$\mathcal{L}_t^{\text{bg}} = \frac{\left\|M^{\text{bg}} \odot (\hat{z}_0 - z_{\text{pre}})\right\|_2^2}{C\left\|M^{\text{bg}}\right\|_1 + \epsilon}, \quad \mathcal{L}_t^{\text{obj}} = \frac{1}{N_{\text{obj}}^+} \sum_{o \in \mathcal{O}^+} \frac{\left\|M_o^{\text{obj}} \odot (\hat{z}_0 - z_{\text{pre}})\right\|_2^2}{C\left\|M_o^{\text{obj}}\right\|_1 + \epsilon}, \tag{14}$$

where ϵ is a small constant for numerical stability, C is the number of latent channels, $\mathcal{O}^+$ denotes the set of non-empty object masks, and $N_{\text{obj}}^+ = |\mathcal{O}^+|$. The overall test-time objective is:

$$\mathcal{L}_t = \lambda_{\text{bg}}\mathcal{L}_t^{\text{bg}} + \lambda_{\text{obj}}\mathcal{L}_t^{\text{obj}}, \tag{15}$$

where λ_{bg} and λ_{obj} balance the background and object supervision. If a supervision mask is empty, its corresponding loss term is set to zero and contributes no gradient.

For each guided denoising step, we hold z_t fixed and perform n_{iter} AdamW updates on ϕ while keeping θ fixed:

$$\phi^{(j+1)} = \text{AdamWStep}\left(\phi^{(j)}, \nabla_\phi \mathcal{L}_t\left(\phi^{(j)}\right); \eta\right), \qquad j = 0, \ldots, n_{\text{iter}} - 1, \tag{16}$$

where AdamWStep denotes one optimizer update, η is the test-time learning rate, and n_{iter} is the number of optimization iterations. The optimized parameters ϕ^* are then used for the corresponding denoising transition,

$$z_{t_{n+1}} = \mathcal{F}_{t_n}\left(z_{t_n}; \theta, \phi^*\right), \tag{17}$$

where n indexes the solver steps with $t_0 = 1$ being the noisiest timestep, and $\mathcal{F}_{t_n}$ denotes the standard diffusion transition induced by the flow-matching scheduler.

This optimization is applied only to the first k solver steps ($n < k$), corresponding to the highest-noise portion of the denoising trajectory where global scene structure is established. The LoRA parameters and optimizer state are carried over across these guided steps, so each step starts from the adapters optimized at the previous step. During the guided steps, classifier-free guidance is disabled and the conditional prediction is used directly for both optimization and the denoising transition. Afterward, the LoRA adapters are explicitly disabled, and the remaining denoising steps proceed with the frozen pretrained backbone and standard classifier-free guidance.

3 Experiments

3.1 Experimental Setup

Implementation Details. Unless otherwise specified, we use the Wan2.2-5B image-to-video model (Wan et al., 2025) as the backbone and generate 81-frame clips at a resolution of 1280×704. During inference, we use 50 denoising steps, apply test-time optimization during the first five steps with four inner-loop updates per step, and use classifier-free guidance (CFG) with a guidance scale of 6.0 for the remaining steps. We use AdamW (Kingma & Ba, 2014) with a learning rate of 10^{-3} and mixed-precision arithmetic during optimization. Both λ_{obj} and λ_{bg} are set to 1.0. We run GT2 on two 48GB NVIDIA RTX 4090 GPUs with FSDP. The peak GPU memory usage is approximately 40GB, and each inference takes about 612 seconds. Additional details are provided in Appendix B.



Figure 3: Qualitative comparison on camera control. In these examples, CameraCtrl and WorldForge tend to produce overly large viewpoint changes while keeping the subjects largely static, whereas ViewCrafter shows noticeable visual inconsistencies. GT2 better follows the commanded motion while preserving scene dynamics and visual quality.

Evaluation Datasets and Metrics. To our knowledge, no public benchmark directly evaluates joint 3D camera and object control. We therefore curate a 100-sample evaluation set spanning diverse camera trajectories and object translations. We evaluate visual quality with VBench++ (Huang et al., 2025b), camera controllability with rotation error (RotErr) and translation error (TransErr) adapted from CameraCtrl (He et al., 2025a), and object controllability with ObjMC following VerseCrafter (Zheng et al., 2026b). RotErr and TransErr are computed as per-frame RMSE over consecutive relative poses, while ObjMC measures the average distance between estimated and target 3D object trajectories. Details of the dataset and metrics are provided in Appendix C and Appendix D.

3.2 Performance Comparison

Following prior works (Zheng et al., 2026b), we evaluate two settings: independent camera control and joint camera–object control. We further include object-centric comparisons in Appendix G.

Table 1: System-level comparison on camera-only control (top) and joint camera–object control (bottom). Higher is better for VBench++ scores; lower is better for RotErr, TransErr, and ObjMC.

Model	Overall Score	Imaging Quality	Aesthetic Quality	Motion Smoothness	Background Consistency	Subject Consistency	I2V Background	I2V Subject	RotErr (deg)	TransErr (m)	ObjMC (m)
CameraCtrl	85.88	62.30	56.05	98.08	**95.22**	94.71	97.46	97.35	0.195	0.134	-
ViewCrafter	85.01	66.65	57.41	97.05	91.86	90.22	95.89	96.00	0.140	0.049	-
WorldForge	86.50	67.14	56.62	97.49	93.42	93.68	98.58	98.59	0.096	0.032	-
GT2 (Camera-only)	**87.68**	**68.73**	**59.20**	**98.62**	94.97	**94.72**	**98.80**	**98.70**	**0.081**	**0.023**	-
PAC	87.34	66.69	**60.74**	97.69	94.96	**95.97**	97.89	97.44	0.156	0.131	2.490
VerseCrafter	87.37	68.75	58.82	98.56	94.52	93.35	**98.77**	98.80	0.104	0.051	1.734
Symphomotion	87.01	67.25	57.69	**98.75**	94.69	94.51	98.14	98.04	0.186	0.045	1.803
GT2 (Joint)	**87.80**	**69.56**	59.34	98.39	**95.03**	94.63	**98.77**	**98.88**	**0.091**	**0.032**	**1.574**

Independent Camera Motion Control. For independent camera control, we use the same 100-sample evaluation set after removing the prescribed object motions. We retain only the background projection, coarsely fill its holes, and optimize using $\mathcal{L}_t = \mathcal{L}_t^{\text{bg}}$. In this setting, object motion is not explicitly constrained, so segmented objects can follow the pretrained generator's motion prior. We compare GT2 with CameraCtrl (He et al., 2025a), ViewCrafter (Yu et al., 2024), and WorldForge (Song et al., 2026) in Fig. 3 and Table 1. GT2 achieves the lowest camera errors and the highest overall VBench++ score among these camera-control baselines.

Joint Camera and Object Motion Control. We compare GT2 with PAC (Chen et al., 2025b), Symphomotion (Zhang et al., 2026), and VerseCrafter (Zheng et al., 2026b) on joint control. As shown in Table 1, GT2 achieves the lowest RotErr, TransErr, and ObjMC among the evaluated methods while maintaining comparable visual-quality scores. Figure 4 provides representative qualitative examples, and multi-seed results are reported in Appendix F.

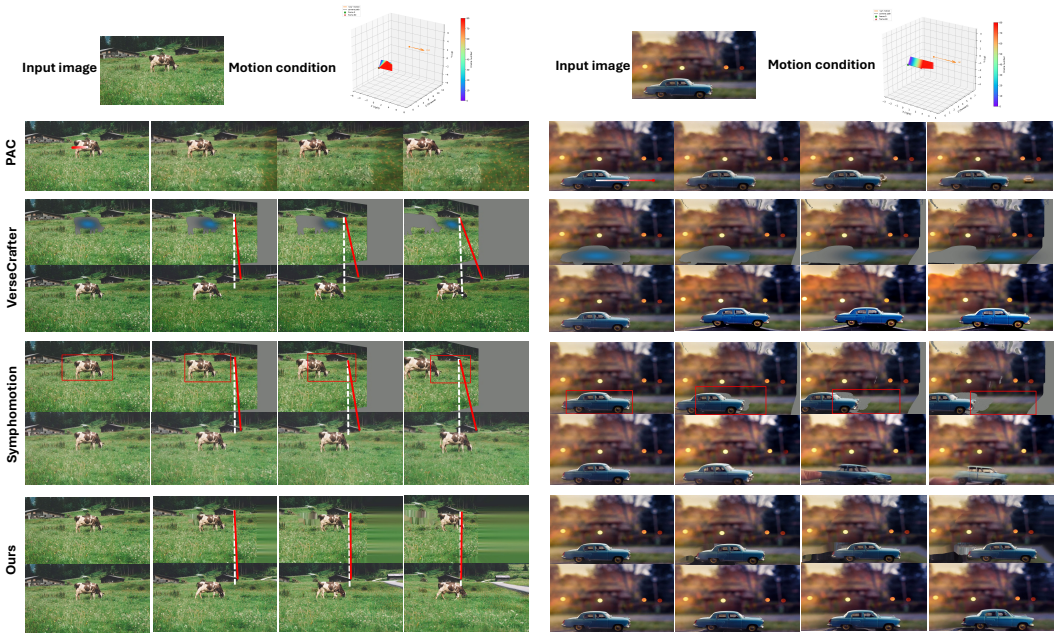

Figure 4: Qualitative comparison on joint camera–object control. In these examples, PAC struggles to follow the commanded object motion, while VerseCrafter and Symphomotion show noticeable visual inconsistencies. GT2 better preserves visual quality while following the prescribed motions.

3.3 Ablation Studies

Effect of Test-Time Optimization. To isolate the contribution of test-time optimization, we compare GT2 with the same Wan2.2 backbone without test-time optimization. The vanilla model receives the input image and the same text prompt describing the video content and intended camera/object motions, so motion information is provided only through language. Table 2 shows that test-time optimization substantially reduces camera and object control errors with only a small change in the Overall Score.

Table 2: Comparison with the Wan2.2 backbone without test-time optimization.

Method	Overall Score	RotErr (deg)	TransErr (m)	ObjMC (m)
Wan2.2 w/o test-time optimization	**88.07**	0.203	0.072	2.154
GT2	87.80	**0.091**	**0.032**	**1.574**

Direct Latent Replacement versus Test-Time Optimization. A simple alternative is to replace the target region of the estimated $\hat{z}_0$ with the preview latent at each guidance step. Table 3 indicates that direct replacement can preserve competitive VBench++ scores, but it yields larger motion errors. Figure 5 further shows that direct replacement may duplicate subjects or fail to follow the commanded motion, suggesting that optimization provides a better control-quality trade-off.

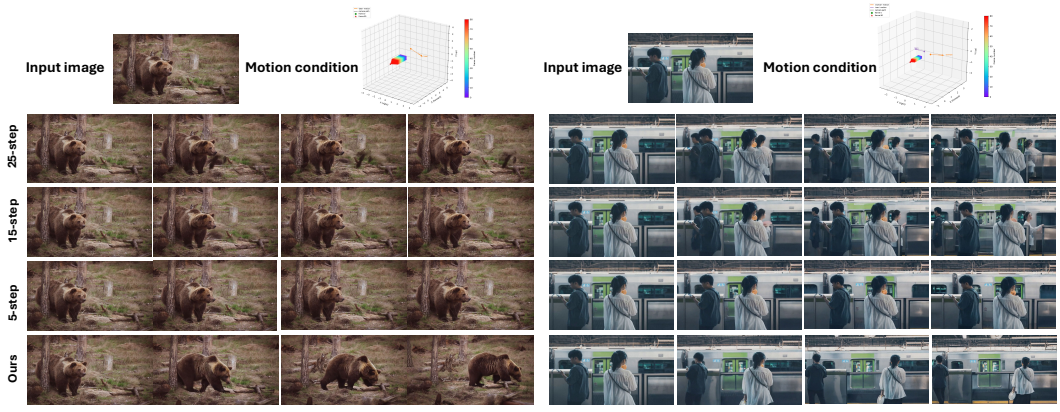

Figure 5: Qualitative comparison of GT2 and direct latent replacement with different guided steps.

Table 3: Comparison between direct latent replacement and test-time optimization.

Strategy	Method	Overall Score	RotErr (deg)	TransErr (m)	ObjMC (m)
Direct Replacement	5 timesteps	**88.05**	0.128	0.046	2.052
	15 timesteps	87.75	0.140	0.051	2.339
	25 timesteps	87.49	0.155	0.071	2.677
Test-time Optimization	GT2 (5 timesteps)	87.80	**0.091**	**0.032**	**1.574**

Guidance Timesteps and Optimization Iterations. Motivated by Qu et al. (2026), we apply test-time optimization only during early denoising. Table 4 varies the number of guided steps and inner-loop updates, with inference latency measured on two 48GB NVIDIA RTX 4090 GPUs using FSDP. Increasing the number of guided steps or optimization iterations can improve individual metrics, but the gains are not consistent across all metrics and come with substantial latency overhead. We therefore adopt 5 guided timesteps and 4 inner-loop iterations as a favorable trade-off between controllability and inference cost.

Table 4: Ablation on the numbers of guided solver steps and inner-loop optimization iterations.

Group	Method	Overall Score	RotErr (deg)	TransErr (m)	ObjMC (m)	Inference Latency (s)
Fixed iterations=4	3 timesteps	**87.89**	0.093	0.042	1.926	**521**
	5 timesteps	87.80	**0.091**	**0.032**	1.574	612
	7 timesteps	87.76	0.094	0.035	**1.460**	711
Fixed timesteps=5	4 iterations	**87.80**	0.091	**0.032**	1.574	**612**
	8 iterations	87.79	**0.077**	0.037	1.535	890
	12 iterations	87.35	0.113	0.059	**1.508**	1163

Component and Supervision Ablations. Projection holes can contaminate neighboring features after VAE encoding and thereby degrade guided generation. Figure 6 illustrates how depth-aware coarse inpainting provides cleaner latent guidance and improves generation quality. Table 5 ablates the geometry-processing components and supervision terms. Removing depth-aware inpainting leads to the largest quality drop, while removing background erosion or object-core selection weakens object control. For the supervision terms, removing the background loss degrades both camera and object control, while removing the object loss improves camera translation at the cost of substantially worse object control. These results show that the two losses provide complementary guidance, with the full objective better balancing camera and object controllability.

Table 5: Ablation study of different components and supervision terms.

Method	Overall Score	RotErr (deg)	TransErr (m)	ObjMC (m)
w/o depth-aware inpainting	86.59	0.109	0.036	1.879
w/o background mask erosion	87.23	0.095	0.032	1.667
w/o object Gaussian-core selection	87.74	0.093	0.031	1.626
w/o background loss	86.87	0.199	0.043	1.884
w/o object loss	**87.83**	0.096	**0.022**	2.176
GT2	87.80	**0.091**	0.032	**1.574**

4 Related Work

Independent Camera or Object Control. Existing controllable video generation methods primarily address camera or object control separately. For camera control, early methods (Wang et al., 2024b; He et al., 2025a) directly encode camera poses into pretrained video diffusion models, while later works (Zheng et al., 2024; Yu et al., 2024; Ren et al., 2025; Cao et al., 2025) introduce stronger geometric priors to train a geometry-conditioned generation model. Training-free approaches (Hou & Chen, 2024; Zhou et al., 2025; Song et al., 2026) instead directly manipulate intermediate latents based on constructed views, avoiding additional task-specific training. For object control, previous works rely mainly on 2D motion cues, including optical flow, point trajectories, masks, and bounding boxes (Yin et al., 2023; Zhang et al., 2025b; Li et al., 2025a; Wang et al., 2024a), while more recent methods incorporate depth or explicit 3D trajectories to enable more flexible motion patterns

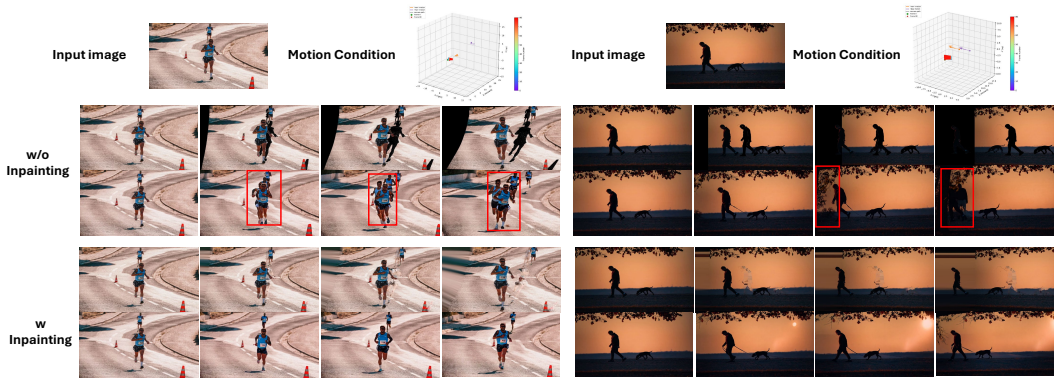

Figure 6: Depth-aware inpainting reduces projection-hole artifacts and improves generation quality.

and improved geometric consistency (Wang et al., 2025a; Fu et al., 2025). While these methods provide increasingly rich control interfaces, their control formulations are typically limited to either camera motion or object motion, rather than jointly modeling both within a shared 3D scene.

Joint Camera-Object Control. Recent works further explore joint camera and object control in video generation. MotionPrompting (Geng et al., 2025), CineMaster (Wang et al., 2025b), and VerseCrafter (Zheng et al., 2026b) couple camera and object dynamics through shared motion priors, while Perception-as-Control (Chen et al., 2025b), VidCraft3 (Zheng et al., 2026a), and Symphomotion (Zhang et al., 2026) introduce more explicit mechanisms to disentangle the two motion types. Nevertheless, these approaches rely on offline training with large-scale datasets and detailed motion annotations, e.g., geometry-based datasets sourced from Sekai (Li et al., 2025b)/SpatialVID (Wang et al., 2026) for VerseCrafter and RealCOD-25K Zhang et al. (2026) for Symphomotion. In contrast, GT2 performs test-time optimization on a pretrained video generator, avoiding substantial task-specific training at the expense of additional computation during inference.

Test-Time Optimization. Test-time optimization adapts pretrained generative models to individual inputs without task-specific training. TTOM (Qu et al., 2026) explores this paradigm for text-to-video alignment by matching cross-attention maps with prescribed layouts. Zo3T (Zhang et al., 2025a) extends it to controllable image-to-video generation by enforcing representation similarity within prescribed object bounding boxes on U-Net (Ronneberger et al., 2015) upsampling feature maps, while Time-to-Move (Singer et al., 2025) modifies the denoising process through region-dependent dual-clock sampling. However, these methods target either layout or a single motion type (object or camera), rather than jointly specifying camera and object motion. In contrast, GT2 directly optimizes the region-masked discrepancy between the predicted clean latent and a geometry-derived previsualization target that explicitly coordinates the prescribed camera and object motions within a shared 4D world, enabling joint motion control.

5 Conclusion

We present GT2, a geometry-guided test-time optimization framework for joint camera and object control in video world models. GT2 composes prescribed camera motions and object translations in a shared 3D coordinate system to construct a motion-aware geometric preview, then uses reliable regions of this preview to supervise lightweight LoRA updates during early denoising steps while keeping the pretrained generator backbone frozen. Experiments on a curated joint-control benchmark with the same Wan2.2 backbone show that GT2 substantially reduces camera- and object-control errors compared with direct inference while maintaining competitive visual quality. Optimizing against the preview also achieves better controllability than directly replacing latent regions with the preview, supporting the effectiveness of localized geometric supervision.

The main limitation is the additional runtime and memory required by per-video optimization, motivating future work on efficient adaptation strategies that preserve the gains in controllability.

AI USE STATEMENT

We used generative AI tools for language editing, proofreading, and code debugging. All AI-assisted content was reviewed and verified by the authors.

ETHICS STATEMENT

This work involves a human evaluation of generated videos. Participants only provided subjective preferences, and no personally identifiable information was collected. The study poses no known ethical risks beyond those associated with standard visual evaluation.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of our method, experimental settings, and evaluation protocols in the main paper and appendix. We will release the source code, benchmark and relevant evaluation scripts to facilitate reproducibility, subject to the licensing requirements of the underlying models and datasets.

REFERENCES

Sherwin Bahmani, Ivan Skorokhodov, Guocheng Qian, Aliaksandr Siarohin, Willi Menapace, Andrea Tagliasacchi, David B Lindell, and Sergey Tulyakov. Ac3d: Analyzing and improving 3d camera control in video diffusion transformers. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22875–22889. IEEE, 2025.

Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. In *CVPR*, 2025.

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.

Alexey Bochkovskiy, Amaël Delaunoy, Hugo Germain, Marcel Santos, Yichao Zhou, Stephan Richter, and Vladlen Koltun. Depth pro: Sharp monocular metric depth in less than a second. In *International Conference on Learning Representations*, volume 2025, pp. 75602–75637, 2025.

Chenjie Cao, Jingkai Zhou, Shikai Li, Jingyun Liang, Chaohui Yu, Fan Wang, Xiangyang Xue, and Yanwei Fu. Uni3c: Unifying precisely 3d-enhanced camera and human motion controls for video generation. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*, pp. 1–12, 2025.

Guibin Chen, Dixuan Lin, Jiangping Yang, Chunze Lin, Junchen Zhu, Mingyuan Fan, Hao Zhang, Sheng Chen, Zheng Chen, Chengcheng Ma, et al. Skyreels-v2: Infinite-length film generative model. *arXiv preprint arXiv:2504.13074*, 2025a.

Yingjie Chen, Yifang Men, Yuan Yao, Miaomiao Cui, and Liefeng Bo. Perception-as-control: Fine-grained controllable image animation with 3d-aware motion representation. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 14380–14389. IEEE, 2025b.

Xiao Fu, Xian Liu, Xintao Wang, Sida Peng, Menghan Xia, Xiaoyu Shi, Ziyang Yuan, Pengfei Wan, Di Zhang, and Dahua Lin. 3dtrajmaster: Mastering 3d trajectory for multi-entity motion in video generation. In *International Conference on Learning Representations*, volume 2025, pp. 6680–6693, 2025.

Daniel Geng, Charles Herrmann, Junhwa Hur, Forrester Cole, Serena Zhang, Tobias Pfaff, Tatiana Lopez-Guevara, Yusuf Aytar, Michael Rubinstein, Chen Sun, et al. Motion prompting: Controlling video generation with motion trajectories. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–12. IEEE, 2025.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2(3), 2018.

Hao He, Yinghao Xu, Yuwei Guo, Gordon Wetzstein, Bo Dai, Hongsheng Li, and Ceyuan Yang. Cameractrl: Enabling camera control for video diffusion models. In *International Conference on Learning Representations*, volume 2025, pp. 100433–100464, 2025a.

Hao He, Ceyuan Yang, Shanchuan Lin, Yinghao Xu, Meng Wei, Liangke Gui, Qi Zhao, Gordon Wetzstein, Lu Jiang, and Hongsheng Li. Cameractrl ii: Dynamic scene exploration via camera-controlled video diffusion models. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 13416–13426. IEEE, 2025b.

Chen Hou and Zhibo Chen. Training-free camera control for video generation. *arXiv preprint arXiv:2406.10126*, 2024.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Jiahui Huang, Qunjie Zhou, Hesam Rabeti, Aleksandr Korovko, Huan Ling, Xuanchi Ren, Tianchang Shen, Jun Gao, Dmitry Slepichev, Chen-Hsuan Lin, et al. Vipe: Video pose engine for 3d geometric perception. *arXiv preprint arXiv:2508.10934*, 2025a.

Ziqi Huang, Fan Zhang, Xiaojie Xu, Yinan He, Jiashuo Yu, Ziyue Dong, Qianli Ma, Nattapol Chanpaisit, Chenyang Si, Yuming Jiang, et al. Vbench++: Comprehensive and versatile benchmark suite for video generative models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025b.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Jing Yu Koh, Honglak Lee, Yinfei Yang, Jason Baldridge, and Peter Anderson. Pathdreamer: A world model for indoor navigation. In *CVPR*, 2021.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.

Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

Quanhao Li, Zhen Xing, Rui Wang, Hui Zhang, Qi Dai, and Zuxuan Wu. Magicmotion: Controllable video generation with dense-to-sparse trajectory guidance. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 12112–12123. IEEE, 2025a.

Quanhao Li, Zhen Xing, Rui Wang, Haidong Cao, Qi Dai, Daoguo Dong, and Zuxuan Wu. Flashmotion: Few-step controllable video generation with trajectory guidance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8986–8996, 2026.

Zhen Li, Chuanhao Li, Xiaofeng Mao, Shaoheng Lin, Ming Li, Shitian Zhao, Zhaopan Xu, Xinyue Li, Yukang Feng, Jianwen Sun, et al. Sekai: A video dataset towards world exploration. *arXiv preprint arXiv:2506.15675*, 2025b.

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

Jiaxu Miao, Xiaohan Wang, Yu Wu, Wei Li, Xu Zhang, Yunchao Wei, and Yi Yang. Large-scale video panoptic segmentation in the wild: A benchmark. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21001–21011. IEEE, 2022.

Leigang Qu, Ziyang Wang, Na Zheng, Wenjie Wang, Liqiang Nie, and Tat-Seng Chua. Ttom: Test-time optimization and memorization for compositional video generation. In *International Conference on Learning Representations*, volume 2026, pp. 70225–70247, 2026.

Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, et al. Grounded sam: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*, 2024.

Xuanchi Ren, Tianchang Shen, Jiahui Huang, Huan Ling, Yifan Lu, Merlin Nimier-David, Thomas Müller, Alexander Keller, Sanja Fidler, and Jun Gao. Gen3c: 3d-informed world-consistent video generation with precise camera control. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6121–6132. IEEE, 2025.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 10674–10685. ieee, 2022.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

Xiaoyu Shi, Zhaoyang Huang, Fu-Yun Wang, Weikang Bian, Dasong Li, Yi Zhang, Manyuan Zhang, Ka Chun Cheung, Simon See, Hongwei Qin, et al. Motion-i2v: Consistent and controllable image-to-video generation with explicit motion modeling. In *ACM SIGGRAPH 2024 Conference Papers*, pp. 1–11, 2024.

Assaf Singer, Noam Rotstein, Amir Mann, Ron Kimmel, and Or Litany. Time-to-move: Training-free motion controlled video generation via dual-clock denoising. *ArXiv*, abs/2511.08633, 2025.

Chenxi Song, Yanming Yang, Tong Zhao, Ruibo Li, and Chi Zhang. Taming video models for 3d and 4d generation via zero-shot camera control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 40352–40363, 2026.

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.

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.

Hanlin Wang, Hao Ouyang, Qiuyu Wang, Wen Wang, Ka Leong Cheng, Qifeng Chen, Yujun Shen, and Limin Wang. Levitor: 3d trajectory oriented image-to-video synthesis. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12490–12500. IEEE, 2025a.

Jiahao Wang, Yufeng Yuan, Rujie Zheng, Youtian Lin, Jian Gao, Lin-Zhuo Chen, Yajie Bao, Chang Zeng, Yanxi Zhou, Xiao-Xiao Long, et al. Spatialvid: A large-scale video dataset with spatial annotations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 42592–42603, 2026.

Jiawei Wang, Yuchen Zhang, Jiaxin Zou, Yan Zeng, Guoqiang Wei, Liping Yuan, and Hang Li. Boximator: Generating rich and controllable motions for video synthesis. *arXiv preprint arXiv:2402.01566*, 2024a.

Qinghe Wang, Yawen Luo, Xiaoyu Shi, Xu Jia, Huchuan Lu, Tianfan Xue, Xintao Wang, Pengfei Wan, Di Zhang, and Kun Gai. Cinemaster: A 3d-aware and controllable framework for cinematic text-to-video generation. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pp. 1–10, 2025b.

Ruicheng Wang, Sicheng Xu, Yue Dong, Yu Deng, Jianfeng Xiang, Zelong Lv, Guangzhong Sun, Xin Tong, and Jiaolong Yang. Moge-2: Accurate monocular geometry with metric scale and sharp details. *arXiv preprint arXiv:2507.02546*, 2025c.

Zhouxia Wang, Ziyang Yuan, Xintao Wang, Yaowei Li, Tianshui Chen, Menghan Xia, Ping Luo, and Ying Shan. Motionctrl: A unified and flexible motion controller for video generation. In *ACM SIGGRAPH 2024 Conference Papers*, pp. 1–11, 2024b.

Weijia Wu, Zhuang Li, Yuchao Gu, Rui Zhao, Yefei He, David Junhao Zhang, Mike Zheng Shou, Yan Li, Tingting Gao, and Di Zhang. Draganything: Motion control for anything using entity representation. In *European Conference on Computer Vision*, pp. 331–348. Springer, 2024.

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. In *ICLR*, 2025.

Shengming Yin, Chenfei Wu, Jian Liang, Jie Shi, Houqiang Li, Gong Ming, and Nan Duan. Dragnuwa: Fine-grained control in video generation by integrating text, image, and trajectory. *arXiv preprint arXiv:2308.08089*, 2023.

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. *arXiv preprint arXiv:2409.02048*, 2024.

Guiyu Zhang, Yabo Chen, Xunzhi Xiang, Junchao Huang, Zhongyu Wang, and Li Jiang. Symphomotion: Joint control of camera motion and object dynamics for coherent video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11127–11137, 2026.

Ruicheng Zhang, Jun Zhou, Zunnan Xu, Zihao Liu, Jiehui Huang, Mingyang Zhang, Yu Sun, and Xiu Li. Zero-shot 3d-aware trajectory-guided image-to-video generation via test-time training. *arXiv e-prints*, pp. arXiv–2509, 2025a.

Zhenghao Zhang, Junchao Liao, Menghao Li, Zuozhuo Dai, Bingxue Qiu, Siyu Zhu, Long Qin, and Weizhi Wang. Tora: Trajectory-oriented diffusion transformer for video generation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2063–2073. IEEE, 2025b.

Guangcong Zheng, Teng Li, Rui Jiang, Yehao Lu, Tao Wu, and Xi Li. Cami2v: Camera-controlled image-to-video diffusion model. *arXiv preprint arXiv:2410.15957*, 2024.

Sixiao Zheng, Zimian Peng, Yanpeng Zhou, Yi Zhu, Hang Xu, Xiangru Huang, and Yanwei Fu. Vidcraft3: Camera, object, and lighting control for image-to-video generation. *IEEE Transactions on Visualization and Computer Graphics*, 2026a.

Sixiao Zheng, Minghao Yin, Wenbo Hu, Xiaoyu Li, Ying Shan, and Yanwei Fu. Versecrafter: Dynamic realistic video world model with 4d geometric control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 40277–40290, 2026b.

Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*, 2018.

Zhenghong Zhou, Jie An, and Jiebo Luo. Latent-reframe: Enabling camera control for video diffusion models without training. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 12779–12789. IEEE, 2025.

A PRELIMINARIES

Video Diffusion Models. Latent video diffusion models (Rombach et al., 2022) perform generation in a learned latent space, where a pretrained 3D VAE encodes a video X into $z_0 = \mathcal{E}(X)$. Under flow matching (Lipman et al., 2022), the noisy latent is interpolated as

$$z_t = (1-t)z_0 + tz_1, \tag{18}$$

where $z_1 \sim \mathcal{N}(0, I)$ and the corresponding velocity is $v_t = z_1 - z_0$. Given the predicted velocity v_θ, the clean latent is estimated by

$$\hat{z}_0 = z_t - t\, v_\theta(z_t, t, c_\tau, c_I), \tag{19}$$

where c_τ and c_I denote the text and image conditions respectively.

Denoising Process. During inference, we use a 50-step noise schedule obtained by applying a shift to a uniform grid:

$$t_k = \frac{\gamma \bar{t}_k}{1 + (\gamma - 1)\bar{t}_k}, \qquad \bar{t}_k = 1 - \frac{k}{50}, \tag{20}$$

where we set $\gamma = 5.0$. The timesteps are traversed in descending order from noise to clean, with $t_0 = 1$ and $t_{49} = 0.0926$. At each timestep, the model predicts the velocity field and the corresponding clean latent $\hat{z}_0$ is estimated using Eq. 19. The final denoising step implicitly reaches zero noise within the scheduler, although $t = 0$ is not explicitly included in the schedule.

B More Implementation details

Details of the Projection Process. We provide additional implementation details on point filtering, invalid-depth handling, and the shared z-buffer used in our projection process. For both background and object points, a projected point is considered a valid candidate only if its projected coordinates lie within the image boundary and its camera-space depth is positive.

Some pixels in the reference-view depth map contain invalid or missing depth estimates, particularly in regions such as the sky. Instead of discarding these pixels, we treat them as distant background regions by assigning a sufficiently large depth before back-projection. Specifically, let $\mathrm{P}_{99}(D_0^{\mathrm{valid}})$ denote the 99th percentile of the valid depth values in the reference view. We assign invalid-depth pixels a far-field depth

$$D_{\mathrm{far}} = \max\left(10\left(\mathrm{P}_{99}\left(D_0^{\mathrm{valid}}\right) + 10\right), \max\left(D_0^{\mathrm{valid}}\right) + 100\right). \tag{21}$$

The corresponding pixels are then back-projected using D_{far} and treated as background candidates during subsequent projection. The large assigned depth makes their translation-induced image displacement negligible, providing a simple far-field approximation for projection purposes while preventing visually continuous regions, such as the sky, from becoming unnecessary holes.

For each target frame, all valid background and object candidates are jointly rasterized using a shared z-buffer. At each pixel, we retain only the candidate with the smallest positive camera-space depth. The source identity of the retained candidate is recorded simultaneously, which determines the corresponding visibility mask. Thus, V_s^{bg} contains only background pixels visible after object–background occlusion, while each V_s^o contains only pixels where object o remains visible after both object–object and object–background occlusion. Pixels for which no candidate is retained are treated as uncovered regions and are subsequently handled by the coarse inpainting procedure.

This shared rasterization ensures that the projected images, depth maps, and visibility masks are all determined by the same depth test, maintaining consistency between preview composition and the supervision masks used in test-time optimization.

Details in depth-aware coarse inpainting. In the depth-aware coarse inpainting, we distinguish small cracks from the remaining large holes based on the effective filling operation. For each layer, small cracks are identified as pixels newly covered by a 3×3 morphological closing and having a sufficient number of valid pixels in their original 8-connected neighborhood. The background and each object layer are processed independently, with at least 3 and 2 original valid neighbors required for background and object layers, respectively. Each newly covered pixel is filled with the mean color of these valid neighbors, and the corresponding layer mask is updated accordingly.

Depth-aware propagation using only valid background pixels. In each iteration, a boundary pixel is filled when at least two 8-connected background neighbors have consistent depths within a relative tolerance of $\tau = 0.1$. The propagation terminates when no boundary pixel can be filled or when fewer than 2% of the current boundary pixels can be filled in an iteration. Any pixels remaining after propagation(such as pixels without valid neighbors) are finally assigned the color of the nearest valid background pixel using a distance transform.

Details in Supervision Region Selection. For the background supervision region, we remove the pixels covered by any projected object from V_s^{bg}, denote the remaining valid background mask as $\hat{M}_s^{\mathrm{bg}}$, and compute its Euclidean distance transform $\rho_s(\mathbf{u})$, i.e., the distance from each pixel

$\mathbf{u} \in \hat{M}_s^{\mathrm{bg}}$ to the mask boundary. We use a percentile-based erosion rather than a fixed pixel radius. Specifically, the erosion threshold is chosen as the 30th percentile of these distances:

$$r_{s,\mathrm{bg}} = Q_{30}\left(\left\{\rho_s(\mathbf{u}) \mid \mathbf{u} \in \hat{M}_s^{\mathrm{bg}}\right\}\right), \tag{22}$$

where Q_{30} denotes the 30th percentile. The background supervision mask is then

$$M_s^{\mathrm{bg}} = \left\{\mathbf{u} \in \hat{M}_s^{\mathrm{bg}} \mid \rho_s(\mathbf{u}) > r_{s,\mathrm{bg}}\right\}. \tag{23}$$

For each object o, we first estimate a 3D Gaussian from the N_o valid 3D points $\{\mathbf{x}_{o,k}\}_{k=1}^{N_o}$ within its initial segmentation mask:

$$\boldsymbol{\mu}_o = \frac{1}{N_o}\sum_{k=1}^{N_o}\mathbf{x}_{o,k}, \qquad \boldsymbol{\Sigma}_o = \frac{1}{N_o}\sum_{k=1}^{N_o}(\mathbf{x}_{o,k}-\boldsymbol{\mu}_o)(\mathbf{x}_{o,k}-\boldsymbol{\mu}_o)^\top. \tag{24}$$

At frame s, the Gaussian is transformed to the object-relative camera coordinate system as

$$\boldsymbol{\mu}_{s,o}^{\mathrm{cam}} = \mathbf{R}_{s,o}^{\mathrm{rel}}\boldsymbol{\mu}_o + \mathbf{t}_{s,o}^{\mathrm{rel}}, \qquad \boldsymbol{\Sigma}_{s,o}^{\mathrm{cam}} = \mathbf{R}_{s,o}^{\mathrm{rel}}\boldsymbol{\Sigma}_o(\mathbf{R}_{s,o}^{\mathrm{rel}})^\top. \tag{25}$$

Writing $\boldsymbol{\mu}_{s,o}^{\mathrm{cam}} = (x,y,z)^\top$ and the camera intrinsics as $\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$ with focal lengths (f_x, f_y) and principal point (c_x, c_y), the projected Gaussian center is

$$\boldsymbol{\mu}_{s,o}^{2D} = \begin{bmatrix} f_x x/z + c_x \\ f_y y/z + c_y \end{bmatrix}. \tag{26}$$

We approximate the covariance of the projected 2D Gaussian by first-order error propagation through the perspective projection. The Jacobian of the projection at $\boldsymbol{\mu}_{s,o}^{\mathrm{cam}}$ is

$$\mathbf{J}_{s,o} = \begin{bmatrix} f_x/z & 0 & -f_x x/z^2 \\ 0 & f_y/z & -f_y y/z^2 \end{bmatrix}, \tag{27}$$

and the resulting image-plane covariance is

$$\boldsymbol{\Sigma}_{s,o}^{2D} = \mathbf{J}_{s,o}\boldsymbol{\Sigma}_{s,o}^{\mathrm{cam}}\mathbf{J}_{s,o}^\top. \tag{28}$$

The projected Gaussian density is therefore

$$G_{s,o}^{2D}(\mathbf{u}) = \frac{\exp\left(-\frac{1}{2}(\mathbf{u}-\boldsymbol{\mu}_{s,o}^{2D})^\top(\boldsymbol{\Sigma}_{s,o}^{2D}+\epsilon\mathbf{I})^{-1}(\mathbf{u}-\boldsymbol{\mu}_{s,o}^{2D})\right)}{2\pi\sqrt{\det(\boldsymbol{\Sigma}_{s,o}^{2D}+\epsilon\mathbf{I})}}, \tag{29}$$

where $\epsilon\mathbf{I}$ is a small numerical regularizer.

We define the object supervision region as the intersection between the projected object coverage and the normalized high-density Gaussian core:

$$M_{s,o}^{\mathrm{obj}} = \left\{\mathbf{u} \in V_s^o \;\middle|\; \frac{G_{s,o}^{2D}(\mathbf{u})}{\max_{\mathbf{v}} G_{s,o}^{2D}(\mathbf{v}) + \epsilon} > \tau_{\mathrm{obj}}\right\}. \tag{30}$$

We set $\tau_{\mathrm{obj}} = 0.5$ throughout all experiments. If no pixel satisfies the above condition, i.e., the projected Gaussian core is empty, the corresponding object supervision term is skipped for that frame.

Details in VAE Encoding and Mask Alignment. For the previsualization preview X_{pre}, we use the deterministic posterior mean of the Wan2.2 VAE rather than sampling from the posterior. The resulting latent is channel-wise normalized using the fixed 48-channel statistics of Wan2.2, i.e., $z_{\mathrm{pre}}^{(c)} = (\mu^{(c)} - \mu_{\mathrm{VAE}}^{(c)})/\sigma_{\mathrm{VAE}}^{(c)}$, following the official VAE implementation. To align the supervision masks with the latent sequence, we follow Wan2.2's $4\times16\times16$ compression: the initial frame is kept as the first latent timestep, while the remaining frames are grouped as $(1:4), (5:8), \ldots, (4k-3:4k)$ and max-pooled within each group along the temporal dimension. The resulting masks are then resized to the latent spatial resolution using nearest-neighbor interpolation, corresponding to a $16\times$ downsampling along both spatial dimensions (e.g., 1280×704 maps to 80×44). Thus, an 81-frame input is aligned to 21 latent timesteps, with the temporal grouping exactly matching the causal chunking used by the Wan2.2 VAE.

Algorithm 1 Test-Time Optimization and Denoising

Initial latent z_{t_0}, conditions c_τ, c_I, frozen backbone v_θ, guided steps k, optimization iterations n_{iter}
Initialize LoRA parameters ϕ once for the input video such that $\Delta W = 0$
for $n = 0, \ldots, N-1$ **do**
  if $n < k$ **then**
    for $j = 0, \ldots, n_{\text{iter}} - 1$ **do**
      $\hat{z}_0 \leftarrow z_{t_n} - t_n \, v_{\theta,\phi}(z_{t_n}, t_n, c_\tau, c_I)$
      Compute $\mathcal{L}_{t_n}(\hat{z}_0; z_{\text{pre}})$ using Eq. 15
      Update ϕ with one AdamW step on $\mathcal{L}_{t_n}$
    end for
    $z_{t_{n+1}} \leftarrow \mathcal{F}_{t_n}(z_{t_n}; \theta, \phi^*)$
  else
    Disable LoRA: $\phi \leftarrow \varnothing$
    Enable CFG and update $z_{t_{n+1}} \leftarrow \mathcal{F}^{CFG}_{t_n}(z_{t_n}; \theta)$
  end if
end for

Details in Test-time optimization. The overall procedure is summarized in Algorithm 1. In this stage, we reinitialize the LoRA for each individual video. For each LoRA parameter $\Delta W = BA$, the factor A is initialized with Kaiming-uniform initialization and B with zeros, resulting in $\Delta W = 0$ at initialization. Within each video, both the LoRA parameters and optimizer state are retained across the guided denoising steps. We set the LoRA rank to 32 and alpha to 64. The weights are injected into the Query, Key, Value, and Output projection layers of each attention block in the video backbone. During inference, we use a flow-matching scheduler with timestep shift $\gamma = 5.0$ and a first-order Euler solver over $N = 50$ steps; the final transition is taken to $\sigma = 0$. We use mixed-precision arithmetic where parameters and activations are bfloat16 with fp32 gradient reduction; the input latents and noise are materialized in bfloat16 as well. We use AdamW (Kingma & Ba, 2014) with a learning rate of 10^{-3}, $\beta = (0.9, 0.999)$, and decoupled weight decay 10^{-2}. Gradients are clipped by global ℓ_2 norm to 10.0 before each update. The reported GPU peak memory is per GPU and the 612-second latency is averaged over 5 runs where the model loading, preview construction and VAE encoding/decoding latency are excluded (Table 4 utilizes the same latency protocol).

C Details on the Evaluation Dataset

To evaluate joint camera–object controllability under diverse visual and motion conditions, we construct a curated benchmark of 100 image-based samples with basic statistics shown in Table 6. The benchmark combines *real-world images*, *movie frames*, and *AI-generated images*, covering diverse visual styles, scene types, and object categories. Rather than sampling from a single homogeneous source, we intentionally curate samples with varied object configurations, object scales, and camera/object motion magnitudes. Each sample is paired with a natural-language caption and a manually specified 3D control condition.

Relation to existing datasets and benchmarks. Recent works have introduced resources with joint camera–object annotations for controllable video generation. VerseCrafter (Zheng et al., 2026b) constructs VerseControl4D with camera trajectories and multi-object 3D Gaussian trajectories, while SymphoMotion (Zhang et al., 2026) introduces RealCOD-25K with paired camera poses and object-level 3D trajectories. However, the publicly released resources do not provide the exact evaluation samples or manifests used for reproducible quantitative comparison. The released repository of VerseControl4D provide the annotation pipeline without clarifications on the evaluation split they use. SymphoMotion likewise releases its training dataset, but does not provide a corresponding evaluation manifest.

Existing single-axis benchmarks are also insufficient for our setting. VIPSeg (Miao et al., 2022) provides dense video object masks for panoptic segmentation, but does not specify 3D object trajectories and camera trajectories as generation controls. RealEstate10K (Zhou et al., 2018) provides camera pose trajectories and a public train/test split, but does not contain object-motion controls and

is strongly biased toward static scenes. Thus, these resources cannot directly evaluate whether a video generation method simultaneously follows prescribed 3D camera and object motions from a source image.

We therefore construct a fixed evaluation benchmark in which each sample starts from a single source image and is paired with explicitly prescribed camera and object trajectories. The identical numerical control conditions are provided to all compared methods, enabling reproducible cross-method evaluation of joint 3D camera–object control.

Data sources and benchmark construction. The final benchmark contains 50 real-world images, 30 movie frames, and 20 AI-generated images. The three groups are curated separately and then combined into the final evaluation set. Real-world images are collected from heterogeneous publicly accessible image sources and selected to provide diversity in scene type, object category, object scale, and spatial configuration. Movie samples are selected from representative frames of diverse shots using the same visual coverage criteria. AI-generated samples are also collected from the public websites such as Civitai.

Table 6: Basic statistics of the evaluation benchmark.

Property	**Value**
Number of samples	100
Image resolution	1280×704
Control sequence length	81 frames

3D control annotation and trajectory construction. Given a source image I_0, we first estimate the scene geometry and camera intrinsics and identify the objects selected for control. The camera trajectory and per-object motions are then specified using our self-curated annotation interface.

For each controllable object o, the annotator interactively drags its centric point in the reconstructed 3D scene to specify its desired spatial motion. The interface records the resulting 3D trajectory, and the recorded path is temporally sampled at S uniformly ordered frames. We represent the object condition as a per-object displacement trajectory

$$\{\delta_o(s)\}_{s=0}^{S-1}, \qquad \delta_o(0) = \mathbf{0}$$

where $\delta_o(s)$ denotes the 3D displacement of object o relative to its initial position. If $\mathbf{p}_o^0$ denotes the object's initial 3D position, the prescribed position at frame s is

$$\mathbf{p}_o(s) = \mathbf{p}_o^0 + \delta_o(s).$$

Thus, the object control used by our method is not a single endpoint translation. Instead, it is the complete S-frame displacement trajectory obtained from the annotated drag path, including the intermediate states between the initial and final positions. For the motion statistics reported below, *3D displacement magnitude* denotes the total endpoint displacement

$$\|\delta_o(S-1)\|_2.$$

This quantity measures the magnitude of the prescribed object motion and is distinct from the object's absolute 3D position.

For camera trajectories, we define the camera in a shared coordinate system with the objects and independently control the translation and rotation directions and magnitudes. Their combinations yield a diverse set of camera trajectories spanning a broad range of motion patterns.

Overall, the benchmark provides a source image together with its prescribed camera and object control conditions. No ground-truth video is generated as part of the benchmark. Instead, the annotated controls serve as the reference conditions that all evaluated methods are asked to follow. Each

method receives the same source image and the corresponding control specification and generates its own control conditions and the resulting videos.

Scene and object distribution. The benchmark covers diverse object configurations, object categories, and scene types. The distributions are summarized in Table 7.

Table 7: Scene and object distribution of the benchmark.

Object configuration	Distribution
Single-object	74%
Multi-object	26%

Object category	Distribution
Animal	41.7%
Person	30.2%
Vehicle	22.3%
Other	5.8%

The benchmark contains 58 nature/landscape scenes, 25 urban/street scenes, and 17 indoor scenes. Object scales also vary substantially. At the reference frame, the median object bounding box is 206×181 pixels, with the 25th and 75th percentiles at 99×128 and 348×251 pixels, respectively. The projected object area has a median of 2.96% of the image, ranging from 0.04% to 16.2%.

Motion coverage. The benchmark is designed to cover a broad range of 3D object and camera motions, from relatively subtle displacements to large motions. Table 8 summarizes the principal motion statistics. Object displacement denotes $\|\delta_o(S-1)\|_2$ over the S-frame control sequence. Camera-center travel denotes the realized 3D displacement implied by the prescribed camera trajectory. Image-space quantities are measured between the first and last frames of the corresponding control sequence.

Table 8: Motion statistics of the evaluation benchmark. Object displacement denotes $\|\delta_o(S-1)\|_2$ over the 81-frame control sequence. Camera-center travel denotes the realized 3D displacement of the prescribed camera trajectory. Image-space quantities are measured between the first and last frames.

Motion	Quantity	Min	Median	Max	Unit
Object	3D displacement magnitude	0.8	3.0	10.4	m
	Centroid excursion	0	235	796	px
Camera	Total rotation	0	13.5	30	deg
	Camera-center travel	0	1.4	5.0	m
	Background global translation	13	222	862	px

For object motion, the prescribed 3D displacement magnitude ranges from 0.8 to 10.4 m, with a median of 3.0 m, and covers diverse directions. These 3D motions correspond to substantial and heterogeneous image-space changes: the median object-centroid excursion is 235 pixels, with a maximum of 796 pixels.

For camera motion, the total rotation ranges from 0° to 30°, with a median of 13.5°. The prescribed camera-center travel ranges from 0 to 5.0 m, with a median of 1.4 m. These trajectories produce substantial changes in the image plane. To quantify the global translational component of this image-space change, we additionally estimate the magnitude of the 2D global translation between the first and last background-only projections using phase correlation. This quantity has a median of 222 pixels and reaches 862 pixels. It should be interpreted as a global image-translation measure rather than a complete measure of camera-induced image motion; in particular, forward/backward camera motion can primarily manifest as scale change rather than translation.

The benchmark therefore includes both independent and coupled camera motions rather than concentrating on a single trajectory type. The median object-centroid excursion is 235 pixels, while the median global background translation is 222 pixels, indicating that the two control components are intentionally configured to produce comparable image-space changes.

Benchmark representativeness. The benchmark is designed as a controlled stress test rather than a statistically representative sample of Internet imagery. Its heterogeneous sources provide variation in visual appearance and scene composition, while the curated distributions provide explicit coverage of single- and multi-object scenes, diverse object categories and scales, and a broad range of camera and object motion magnitudes. The benchmark is intended to complement existing joint camera–object training resources by providing a fixed image-conditioned evaluation protocol in which the prescribed 3D control trajectories are shared across methods.

D Details on the Evaluation Metrics

All evaluation metrics used in our experiments have been adopted by prior approaches (He et al., 2025a; Zhang et al., 2026; Zheng et al., 2026b). Since these metrics rely on estimated geometric quantities from the generated videos, such as camera trajectories and depth, the resulting measurements may be affected by estimation errors. We apply the same geometry estimation and evaluation procedures to all compared methods, ensuring that they are evaluated under a consistent protocol.

D.1 Camera Trajectory Alignment

For camera-motion evaluation, we estimate the camera trajectory of each generated video using VIPE (Huang et al., 2025a). Since the camera trajectory estimated by VIPE may differ from the prescribed trajectory in the MoGe-v2 coordinate system in terms of global coordinate frame and scale, we align the estimated camera trajectory with the prescribed camera trajectory following the trajectory alignment procedure used in prior camera-control evaluation (He et al., 2025b). Specifically, the alignment consists of four steps: (1) centering both trajectories by subtracting their respective means; (2) finding the optimal scale factor between the trajectories; (3) computing the optimal rotation between the centered trajectories using singular value decomposition (SVD); and (4) determining the translation that aligns the trajectories. The resulting transformation is applied to the estimated camera trajectory before computing the camera-motion errors.

D.2 Rotation Error (RotErr)

To assess the consistency between the generated camera motion and the prescribed trajectory, we adopt the relative rotation error metric. For each generated video with S frames, VIPE (Huang et al., 2025a) provides a sequence of camera poses $\{\widetilde{\mathbf{T}}^j_{\text{gen}}\}_{j=1}^{S}$ after the trajectory alignment described above, where each pose $\widetilde{\mathbf{T}}^j_{\text{gen}} = \begin{bmatrix} \widetilde{\mathbf{R}}^j_{\text{gen}} & \widetilde{\mathbf{t}}^j_{\text{gen}} \\ \mathbf{0}^\top & 1 \end{bmatrix} \in SE(3)$ consists of a rotation matrix $\widetilde{\mathbf{R}}^j_{\text{gen}}$ and a translation vector $\widetilde{\mathbf{t}}^j_{\text{gen}}$. The corresponding prescribed camera poses $\{\mathbf{T}^j_{\text{ctrl}}\}_{j=1}^{S}$ are defined analogously with rotations $\mathbf{R}^j_{\text{ctrl}}$ and translations $\mathbf{t}^j_{\text{ctrl}}$.

We first compute the relative rotations between consecutive frames:

$$\Delta\mathbf{R}^j_{\text{ctrl}} = \left(\mathbf{R}^j_{\text{ctrl}}\right)^{-1}\mathbf{R}^{j+1}_{\text{ctrl}}, \qquad \Delta\widetilde{\mathbf{R}}^j_{\text{gen}} = \left(\widetilde{\mathbf{R}}^j_{\text{gen}}\right)^{-1}\widetilde{\mathbf{R}}^{j+1}_{\text{gen}}. \tag{31}$$

The rotational discrepancy is measured by the RMSE of the relative angular errors:

$$\text{RotErr} = \sqrt{\frac{1}{S-1}\sum_{j=1}^{S-1}\left[\arccos\left(\frac{\text{tr}\left(\left(\Delta\mathbf{R}^j_{\text{ctrl}}\right)^\top\Delta\widetilde{\mathbf{R}}^j_{\text{gen}}\right)-1}{2}\right)\frac{180}{\pi}\right]^2}. \tag{32}$$

where $\text{tr}(\cdot)$ denotes the matrix trace. A lower RotErr indicates closer agreement between the generated and prescribed relative camera rotations.

D.3 Translation Error (TransErr)

We further evaluate the consistency of the generated camera trajectory in terms of relative translation. Using the aligned camera poses, we compute the relative camera motions between consecutive

frames as

$$\Delta\widetilde{\mathbf{T}}^{j}_{\text{gen}} = \left(\widetilde{\mathbf{T}}^{j}_{\text{gen}}\right)^{-1} \widetilde{\mathbf{T}}^{j+1}_{\text{gen}}, \qquad \Delta\mathbf{T}^{j}_{\text{ctrl}} = \left(\mathbf{T}^{j}_{\text{ctrl}}\right)^{-1} \mathbf{T}^{j+1}_{\text{ctrl}}, \tag{33}$$

and denote by $\Delta\widetilde{\mathbf{t}}^{j}_{\text{gen}}$ and $\Delta\mathbf{t}^{j}_{\text{ctrl}}$ the translation components of the relative poses $\Delta\widetilde{\mathbf{T}}^{j}_{\text{gen}}$ and $\Delta\mathbf{T}^{j}_{\text{ctrl}}$, respectively. We define the translation error as the RMSE of the relative translation discrepancies:

$$\text{TransErr} = \sqrt{\frac{1}{S-1}\sum_{j=1}^{S-1}\left\|\Delta\mathbf{t}^{j}_{\text{ctrl}} - \Delta\widetilde{\mathbf{t}}^{j}_{\text{gen}}\right\|_2^2}, \tag{34}$$

where each term measures the deviation in relative camera translation between consecutive frames. A lower TransErr indicates closer agreement with the prescribed camera motion.

D.4 Object Motion Control (ObjMC)

Following the 3D Gaussian trajectory representation used in VerseCrafter (Zheng et al., 2026b), we evaluate object-motion controllability by measuring the discrepancy between the reconstructed and prescribed 3D object trajectories in a common world coordinate system. For each generated video, we first estimate the camera trajectory using VIPE (Huang et al., 2025a) and align it with the prescribed camera trajectory using the similarity transformation described above.

For each generated frame, we segment the controlled objects using Grounded-SAM2 and estimate the depth and camera intrinsics using MoGe-v2. The depth values within each object mask are back-projected into 3D, and a 3D Gaussian is fitted to the resulting points, yielding a Gaussian mean in the corresponding camera coordinate system. We then transform the reconstructed Gaussian means into the world coordinate system using the aligned camera poses.

Let N_{ctrl} and N_{pred} denote the number of prescribed and predicted controlled objects, respectively, and let T be the number of evaluated frames. For prescribed object $o \in \{1, \dots, N_{\text{ctrl}}\}$ and predicted object $k \in \{1, \dots, N_{\text{pred}}\}$, we denote their 3D Gaussian means at frame t in the aligned world coordinate system by $\boldsymbol{\mu}_o^{(t)} \in \mathbb{R}^3$ and $\hat{\boldsymbol{\mu}}_k^{(t)} \in \mathbb{R}^3$, respectively.

Trajectory distance and matching. We first measure the discrepancy between a pair of object trajectories by averaging the Euclidean distance between their 3D Gaussian means over all evaluated frames:

$$d(o,k) = \frac{1}{T}\sum_{t=1}^{T}\left\|\hat{\boldsymbol{\mu}}_k^{(t)} - \boldsymbol{\mu}_o^{(t)}\right\|_2. \tag{35}$$

Using these pairwise distances, we construct a cost matrix $\mathbf{C} \in \mathbb{R}^{N_{\text{ctrl}} \times N_{\text{pred}}}$ with $C_{ok} = d(o,k)$. Since the numbers of predicted and prescribed objects may differ, we augment the cost matrix with dummy rows and columns and assign a fixed penalty λ to dummy matches. We set $\lambda = 10.0\,\text{m}$ in all experiments. The Hungarian algorithm (Kuhn, 1955) is then used to obtain the minimum-cost one-to-one assignment. Each prescribed object is therefore either matched to one predicted object or assigned to a dummy entry.

ObjMC score. Given the resulting assignment, the error associated with each prescribed object o is

$$d_o = \begin{cases} d(o,k) & \text{if } o \text{ is matched to a predicted object } k, \\ \lambda & \text{if } o \text{ is unmatched.} \end{cases} \tag{36}$$

The final ObjMC score is defined as the average trajectory error across all prescribed controlled objects:

$$\text{ObjMC} = \frac{1}{N_{\text{ctrl}}}\sum_{o=1}^{N_{\text{ctrl}}} d_o. \tag{37}$$

A lower ObjMC score indicates that the reconstructed object trajectories more closely follow the prescribed 3D trajectories, while the penalty λ explicitly accounts for missing object predictions.

E Details on the Comparison with Baselines

Table 1 presents a system-level comparison across different control settings and backbone architectures. We rerun each baseline using its latest officially released implementation and convert the prescribed 3D motion conditions into the native control inputs required by each method. The corresponding backbone, output resolution, and frame count are summarized in Table 9.

Table 9: Baseline settings.

Model	Backbone	Resolution	Frame count
CameraCtrl	SVD	576x320	14
ViewCrafter	SVD	1024x576	25
WorldForge	Wan(5B)	1280x704	81
GT2 (camera-only)	Wan(5B)	1280x704	81
PAC	SVD	768x512	16
Symphomotion	Wan(14B)	832x480	81
VerseCrafter	Wan(14B)	1280x720	81
GT2 (joint)	Wan(5B)	1280x704	81

Despite the system-level nature of the comparison, GT2 consistently achieves favorable performance across both control settings as shown in Table 1. For camera-only control, GT2 uses the same backbone, resolution, and frame count as WorldForge, while achieving a higher overall VBench++ score and lower camera control errors. For joint camera–object control, GT2 achieves higher video quality and more accurate control than VerseCrafter and Symphomotion, while using a substantially smaller backbone.

F Multi-Seed Results

To provide a statistical evaluation, we repeat the joint control experiment with five different random seeds. The resulting controllability performance, reported as mean $\pm$ standard deviation (std), is shown in Table 10, which shows that our method achieves consistently better performance than the baselines across all three metrics, with limited variation across random seeds.

Table 10: Multi-seed results.

Model	RotErr	TransErr	ObjMC
PAC	0.159$\pm$0.004	0.129$\pm$0.006	2.462$\pm$0.053
Symphomotion	0.175$\pm$0.008	0.049$\pm$0.003	1.860$\pm$0.062
VerseCrafter	0.098$\pm$0.005	0.051$\pm$0.007	1.696$\pm$0.047
ours	0.092$\pm$0.003	0.033$\pm$0.002	1.536$\pm$0.033

G Comparison with Object-Centric Baselines

We further compare our method with two object-centric methods: DragAnything(Wu et al., 2024) and FlashMotion (Li et al., 2026) on our evaluation set. We use the 2D-level ObjMC used in DragAnything (Wu et al., 2024) to evaluate the object controllability. The results are shown in Table 11. Our method also presents superior performance.

Table 11: Comparison with object-centric baselines

Model	ObjMC-2D(px)
DragAnything	55.23
FlashMotion	50.98
ours	**43.67**

H EXPERIMENTS ON EXTERNAL EVALUATION DATASETS

We further conduct camera-only controllability evaluation on the external Sekai (Li et al., 2025b) dataset, where we randomly select 100 videos as our evaluation dataset. The results are shown in Table 12, and we use FVD (Unterthiner et al., 2018) as the evaluation metric for the video quality. Our method consistently outperforms other baselines in both controllability and video quality.

Table 12: Evaluation on the external Sekai dataset.

Model	RotErr	TransErr	FVD
CameraCtrl	0.471	0.219	692.67
ViewCrafter	0.247	0.082	573.63
WorldForge	0.269	0.095	401.54
Ours	**0.218**	**0.053**	**257.19**

I ADAPTING TO DIFFERENT VIDEO BACKBONES

To demonstrate that our method can be readily adapted to different video backbones, we implement it on another two widely-used video generators: CogvideoX (Yang et al., 2025) and SkyReel-V2 (Chen et al., 2025a). The results are illustrated in Table 13, which further validate the effectiveness of our method.

Table 13: Adapt our test-time optimization method on different video backbones.

Model	Resolution	Frames	Overall Score	RotErr	TransErr	ObjMC
CogvideoX-5B	720x480	49	87.34	0.233	0.073	2.378
CogvideoX-5B with test-time optimization	720x480	49	86.91	0.120	0.066	1.642
SkyReels-V2-1.3B	960x544	97	85.97	0.251	0.109	2.888
SkyReels-V2-1.3B with test-time optimization	960x544	97	86.07	0.113	0.040	1.663

J INFERENCE COST

We further compare the inference cost with the closest baseline, VerseCrafter. As shown in Table 14, our method requires substantially fewer computational resources. Notably, VerseCrafter requires eight 48GB NVIDIA RTX 4090 GPUs with FSDP, sequence parallelism (SP), and CPU offloading to avoid OOM errors. In contrast, our method runs on only two 48GB RTX 4090 GPUs with FSDP, without requiring sequence parallelism or CPU offloading. Our method achieves superior performance with a smaller backbone and substantially lower inference cost.

Table 14: Inference cost comparison between our method and VerseCrafter.

method	backbone	resolution	frames	GPU requirement	peak memory	inference latency(s)
VerseCrafter	Wan(14B)	1280x720	81	8 48G RTX 4090	47GB	> 3600
Ours	Wan(5B)	1280x704	81	2 48G RTX 4090	40GB	612

In addition, we also compare the inference cost of our method with standard inference using CFG. All experiments are conducted on two 48GB NVIDIA RTX 4090 GPUs with FSDP, and the results are reported in Table 15. Our method increases the inference latency by approximately $1.7\times$ and requires higher peak memory, reflecting the additional cost of test-time optimization. Given the added capability of jointly controlling camera and object dynamics, we consider these increased inference cost a worthwhile trade-off.

Table 15: Inference cost comparison between standard inference with CFG and our method.

method	peak memory	inference latency(s)
standard inference with CFG	25GB	360
Ours	40GB	612

K Ablation of Outside-Mask Preservation

To examine whether updating the shared LoRA parameters introduces unintended changes outside the supervised regions, we additionally consider an outside-mask preservation objective. Let $M^{\mathrm{sup}} = M^{\mathrm{bg}} \vee \bigvee_{o\in\mathcal{O}^+} M_o^{\mathrm{obj}}$ denote the union of the background and object supervision masks, and let $M^{\mathrm{out}} = 1 - M^{\mathrm{sup}}$ denote the regions excluded from the control objective. While our default objective only optimizes the supervised regions toward the previsualization target z_{pre}, updating the shared LoRA parameters can nevertheless alter the model prediction in M^{out}. To explicitly suppress such unintended changes, we introduce an additional preservation term:

$$\mathcal{L}_t^{\mathrm{out}} = \frac{\left\| M^{\mathrm{out}} \odot \left(\hat{z}_0^{\mathrm{LoRA}} - \hat{z}_0^{\mathrm{base}} \right) \right\|_2^2}{C \left\| M^{\mathrm{out}} \right\|_1 + \epsilon}, \tag{38}$$

where $\hat{z}_0^{\mathrm{LoRA}}$ denotes the clean-latent prediction from the model with the current LoRA parameters, and $\hat{z}_0^{\mathrm{base}}$ is the corresponding prediction from the frozen backbone without LoRA. The complete objective then becomes

$$\mathcal{L}_t = \lambda_{\mathrm{bg}} \mathcal{L}_t^{\mathrm{bg}} + \lambda_{\mathrm{obj}} \mathcal{L}_t^{\mathrm{obj}} + \lambda_{\mathrm{out}} \mathcal{L}_t^{\mathrm{out}}. \tag{39}$$

Thus, the original background and object losses encourage the optimized prediction to match the prescribed geometric target in the reliable supervised regions, while the additional term acts as a consistency constraint that anchors the remaining regions to the original model prediction.

We set λ_{out} to be 1.0, and perform a qualitative case study by comparing optimization with and without this additional preservation term. As shown in Fig. 7, in both examples, explicitly preserving the outside-mask regions does not provide an apparent improvement in visual quality. In contrast, the variant without the preservation term exhibits stronger adherence to the prescribed camera and object motions. Based on these observations, we omit the explicit outside-mask preservation term from our final objective.

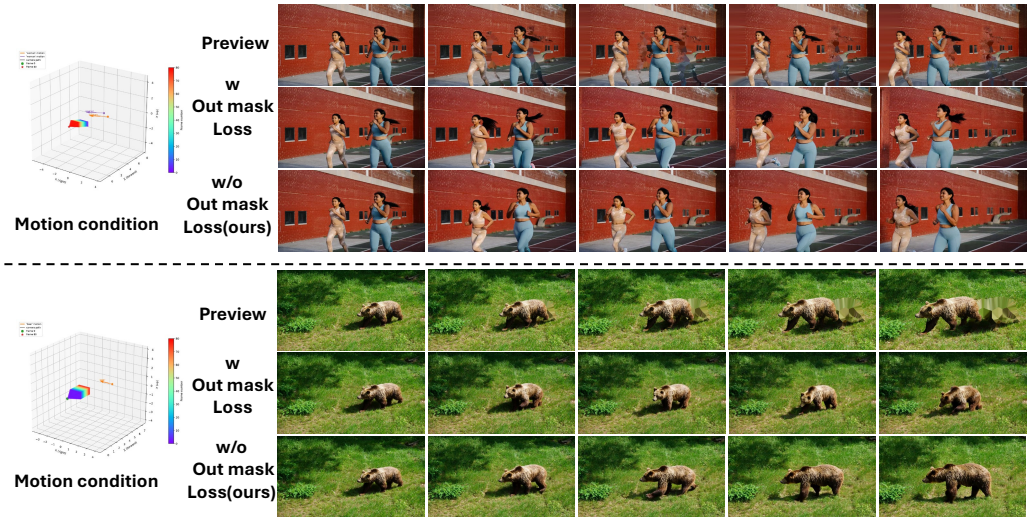

Figure 7: Ablation on the outside-mask preservation.

L Ablation on Depth Estimation Models

Different depth estimation models can introduce different levels of geometric noise. Our coarse inpainting and reliable region selection strategies help mitigate the effect of imperfect depth reconstruction. As shown in Fig. 8, replacing MoGe-v2 with Depth Pro (Bochkovskiy et al., 2025) results in qualitatively stable control across the generated videos. This suggests that the proposed framework is not restricted to a specific monocular depth estimator.

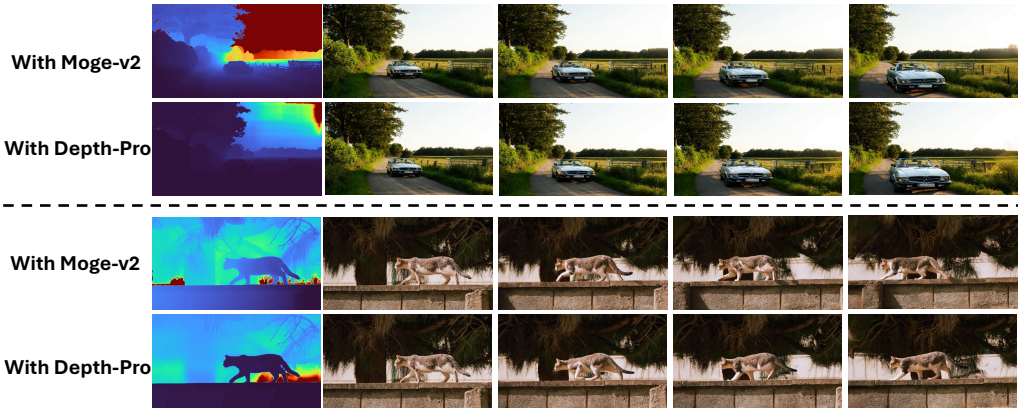

Figure 8: Ablation on depth estimation. Our method remains stable with different depth estimators.

M Human Evaluation

We compare our method with three representative baselines, PAC, Symphomotion, and VerseCrafter, using 25 video pairs for each baseline. In total, 20 human evaluators participate in the study and select, for each pair, the video that better satisfies the prescribed joint camera–object motion while maintaining visual quality. The resulting preference rates are shown in Fig. 9.

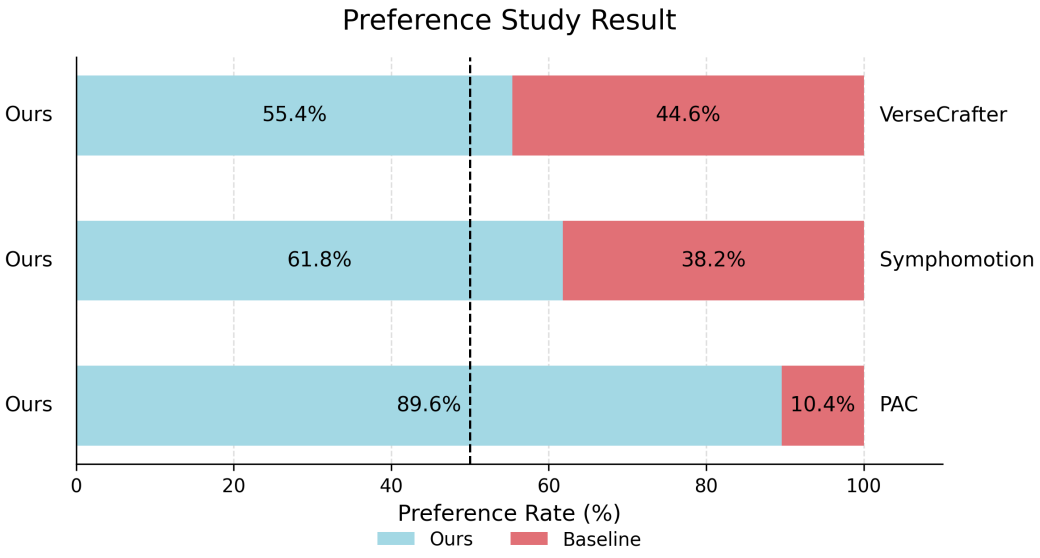

Figure 9: Human preference rates of our method compared with three representative baselines.

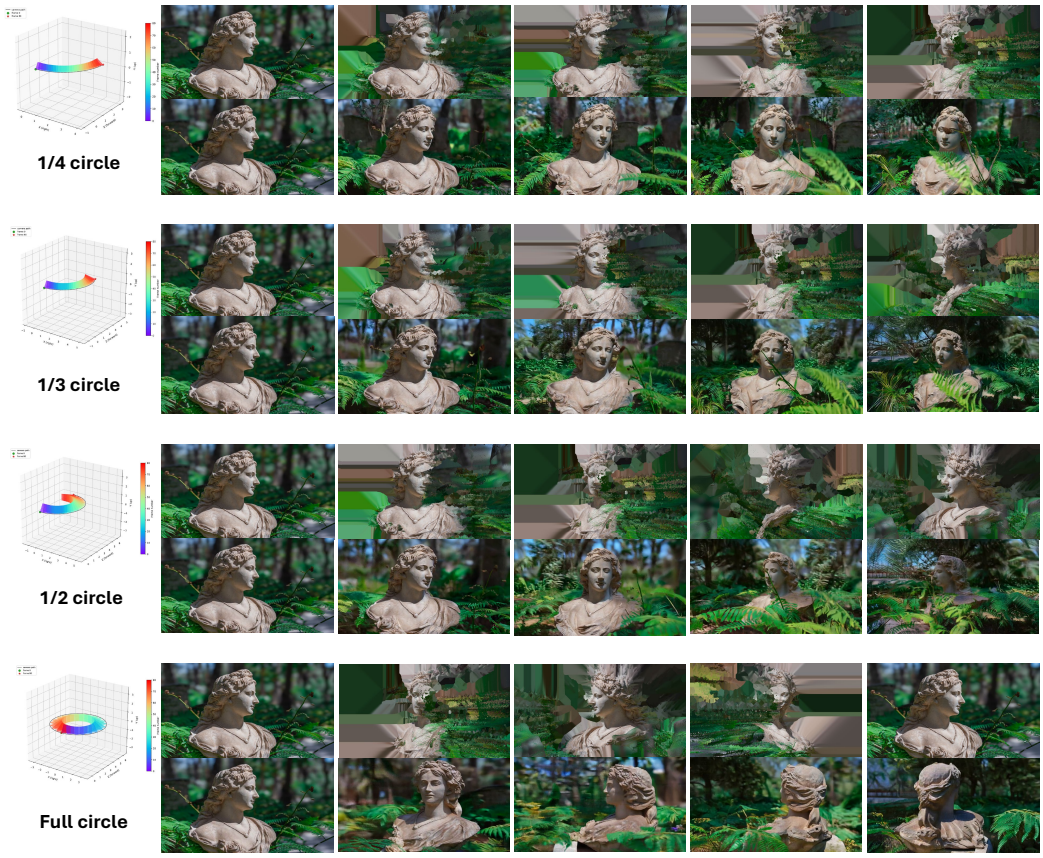

Figure 10: Case Study on Large-Angle Camera Orbits.

N Performance on challenging scenes

Since our method relies on depth estimation and geometric projection, large camera motions can substantially reduce the visible regions that can be reliably reused, which may affect both generation quality and controllability. We conduct a simple case study by specifying orbit motions around the object with angular ranges of a quarter circle (90°), one-third of a circle (120°), a half circle (180°), and a full circle (360°), as shown in Figure 10. As the orbit becomes larger from a quarter circle to a half circle, the generation quality gradually declines, while the generated videos still largely follow the prescribed motion. When a full-circle orbit is specified, however, the model exhibits noticeable deviations from the intended motion and fails to consistently follow the prescribed trajectory.

O Limitations

First, our method relies on the accuracy of depth estimation to construct the 3D-aware guidance. Consequently, errors in the estimated depth can propagate to the subsequent projection and warping processes, leading to degraded control performance. This limitation is also shared by other depth-dependent approaches (Song et al., 2026; Yu et al., 2024).

Second, our method relies on the visible regions of the background and controlled objects in the reference view to construct the projection-based guidance. When only a small portion of the scene or an object is visible, the resulting guidance becomes less informative, which can limit the accuracy and effectiveness of the corresponding motion control.

Third, our current formulation focuses on translational object motion. Although the framework can in principle be extended to more general transformations, such as object rotation, we observe that the control accuracy decreases when more complex object motions are introduced. We leave more expressive and accurate object-motion control, including rotation, to future work.

Finally, compared with direct inference, our test-time optimization-based method incurs higher inference latency and memory consumption. We view this additional computational cost as a trade-off for improved controllability over joint camera and object motions.