# SYNCVID: WORLD-EQUIVARIANT DIFFUSION FOR SYNCHRONIZED VIDEO GENERATION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Generating multiple videos of a dynamic scene from a single reference video requires synthesizing both observed and unobserved content consistently across viewpoints. Existing camera-control methods for video diffusion typically rely on architectural mechanisms, including camera embeddings, cross-view attention, and geometric conditioning. Although effective for static scene content, these approaches often struggle with the fine-grained, dynamic geometry that must remain coherent across views and over time. We synchronize the diffusion process itself by sampling the Gaussian noise prior in a canonical 3D space derived from the source video and rasterizing it into target cameras, rather than sampling independently in each 2D image plane. Our method, SyncVid, lifts a monocular video into a canonical 4D representation, associates noise with scene geometry and unseen space, and rasterizes the resulting structured noise into arbitrary views. Because rendering can introduce local correlations, we further propose a cross-view-consistent whitening procedure that decorrelates neighboring pixels while preserving correspondence across views. The structured noise serves as the diffusion prior for a LoRA-adapted pretrained video diffusion model, requiring no architectural modifications. By enforcing spatiotemporal and cross-view coherence at the source of stochasticity, SyncVid generates synchronized videos and improves both perceptual quality and cross-view consistency in our experiments.

## 1 INTRODUCTION

Video diffusion models have rapidly matured from generating a single plausible video to supporting fine-grained user control, including camera trajectories He et al. (2025); Wang et al. (2024); Xu et al. (2024b). A natural extension of this control is *synchronized* video generation: producing multiple videos of the same dynamic scene from different viewpoints that remain consistent with one another. This capability is central to applications such as virtual reshooting and cinematography, multi-view content creation for AR/VR, and synthesizing training data for downstream 3D and 4D vision tasks. Recent work approaches this problem through camera-conditioned reshooting, self-supervised pseudo-view training, or joint multi-view denoising with cross-view interaction modules Bai et al. (2025a); Paliwal et al. (2026); Bai et al. (2025b); Kuang et al. (2024); Xu et al. (2024a). These methods demonstrate that video diffusion models can produce multiple views with useful visual and geometric agreement, but rely primarily on architectural coupling, rendered conditioning, or specialized training data.

A separate line of research has shown that the noise input to a video diffusion model is itself a powerful conditioning signal, rather than merely an initialization to be denoised away. Correlating noise across frames, instead of sampling it independently, improves temporal consistency and enables direct control over motion. PYoCo Ge et al. (2023) introduces correlated noise priors for video diffusion, while *How I Warped Your Noise* Chang et al. (2024) and *Go-with-the-Flow* Burgert et al. (2025) warp a continuous noise field according to optical flow. EquiVDM Liu & Vahdat (2025) further formalizes the resulting equivariance between noise transformations and generated motion. These results suggest that *structured noise* – noise with an explicit, non-trivial correlation structure tied to the underlying content – can serve as an effective and lightweight conditioning mechanism, complementing changes in model architecture or training data.



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.

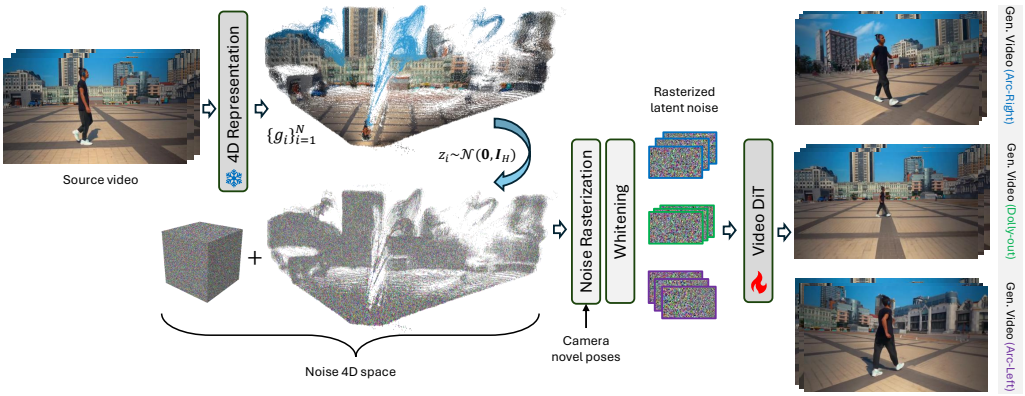


Figure 1: Overview of SycVid. Given a source video, SycVid (1) lifts the observations into a canonical 4D point representation; (2) samples a Gaussian noise vector for each 3D primitive in the scene; and (3), after specifying novel camera trajectories, rasterizes and whitens the shared 3D noise into the corresponding views. These synchronized priors guide video diffusion to produce temporally and cross-view consistent videos from multiple cameras.

Despite this progress, current approaches for constructing structured noise typically warp a 2D noise map using *in-plane* rotations and translations of 2D noise patches to propagate samples into future frames. Alternatively, they require the optical flow of the target video, which is impractical to obtain before generating that video. These limitations can substantially degrade video quality, particularly for dynamic objects undergoing *out-of-plane* rotations, partial occlusions, and subsequent reappearance.

In this work, we show that synchronizing the diffusion prior in a canonical 3D space can induce world-equivariant video generation. Specifically, camera-induced changes in the 3D noise representation are reflected as consistent changes in the generated views. We achieve this by synchronizing the diffusion process at its source of randomness: we construct a canonical 3D noise representation that spans the reconstructed scene observed in the reference video as well as its surrounding unobserved space. We then rasterize this representation into each target camera to produce view-consistent noise priors, including for complex transformations such as out-of-plane rotations. Because all target views derive their stochastic initialization from the same canonical representation, the diffusion model is encouraged to generate correlated completions of unseen content, while conditioning on the reconstructed scene preserves consistency in observed regions. An overview of our method is shown in Fig. 1. We summarize our contributions as follows:

- We propose a structured noise prior for video diffusion models by sampling and assigning noise vectors in the 3D space of the video content, rather than independently in each camera's 2D image plane. Rasterizing this representation preserves dense spatiotemporal structure across views, without relying on 2D warping or editing of image-plane noise.
- We introduce a view-consistent whitening procedure that reduces local spatial correlations introduced by rasterization while preserving cross-view correspondence, keeping the resulting noise close to the marginal distribution expected by the pretrained diffusion model.
- We establish an automated benchmarking pipeline for synchronized video generation on the Mixkit subset of Open-Sora, enabling reliable and reproducible comparisons without manual or subjective selection of evaluation samples.

## 2 Related Work

### 2.1 Structured Noise for Synchronized Video Generation

Structured noise has emerged as an effective mechanism for improving temporal consistency and controlling motion in video diffusion models. PYoCo Ge et al. (2023) observes that when an image diffusion model is adapted into a video model, the noise injected at different frames should preserve temporal correlation rather than being independent: independent per-frame noise lets the denoiser

drift toward a slightly different appearance in each frame, causing jitter and flicker, whereas correlated noise starts neighboring frames from related latent structure and denoises them into consistent appearance and smoother motion. This gain requires no new architecture—no motion modules, optical-flow supervision, or video transformers—but only a modified noise prior, so a pretrained image model reaches strong temporal consistency by fine-tuning on comparatively little video data at low cost. Later work makes the correlation explicit: *How I Warped Your Noise* Chang et al. (2024) transports a continuous Gaussian noise field across frames while preserving its distribution, *Go-with-the-Flow* Burgert et al. (2025) warps noise by real-time optical flow to control object and camera motion, and EquiVDM Liu & Vahdat (2025) shows that training with warped noise encourages equivariance between noise transformations and generated motion. All build correspondence through 2D motion warping, which suits consecutive frames of a single camera, where motion is well approximated by an in-plane flow field, but does not extend across cameras: a 2D warp expresses neither out-of-plane camera rotation nor correspondence for surfaces that a different viewpoint newly reveals or occludes.

### 2.2 Synchronized Camera-Controlled Video Generation

Early camera-control methods such as CameraCtrl He et al. (2025), MotionCtrl Wang et al. (2024), and CamCo Xu et al. (2024b) condition single-view generation on camera trajectories, but do not explicitly enforce cross-view synchronization. Other methods couple denoising across views: SynCamMaster Bai et al. (2025b) adds a multi-view synchronization module for joint multi-camera generation, Collaborative Video Diffusion (CVD) Kuang et al. (2024) exchanges information through epipolar-guided cross-view attention, and CAVIA Xu et al. (2024a) uses spatial and temporal cross-attention for view consistency. A third line uses geometry as an indirect synchronization signal: ReCamMaster Bai et al. (2025a) reshoots videos along novel trajectories using video conditioning and synchronized synthetic multi-view training data, Reshoot-Anything Paliwal et al. (2026) reduces dependence on paired multi-view data through self-supervised pseudo-view training around a 4D point-cloud anchor, and ViewCrafter Yu et al. (2024) guides diffusion-based novel-view video synthesis with coarse rendered geometry. Diffusion-enhancement methods address related but different settings: ArtiFixer De Lutio et al. (2026) couples video diffusion with 3D reconstruction, using opacity-aware latent mixing and autoregressive novel-view generation to enhance incomplete reconstructions, while SetDiff Zanjani et al. (2026) enhances Gaussian-Splatting novel-view renderings in a multi-view setting by set-based diffusion over variable-size reference and target views grounded in pixel-aligned coordinate maps and pose-aware Plucker ray embeddings. Overall, existing methods synchronize through camera embeddings, cross-view interaction modules Bai et al. (2025b); Kuang et al. (2024); Xu et al. (2024a), or rendered geometric conditions Bai et al. (2025a); Paliwal et al. (2026); Yu et al. (2024); Zanjani et al. (2026). In contrast, we impose camera control with structured noise: instead of sampling prior Gaussian noise independently in each camera plane, we sample a canonical 3D field and project it into each view, providing both geometric conditioning and view-consistent diffusion initialization.

## 3 Method

### 3.1 Preliminary: Noise Sampling in Video Diffusion Models

Let $\mathbf{x}_0 \in \mathbb{R}^{t\times C\times h\times w}$ denote a clean VAE latent features of a video $\mathbf{v} \in \mathbb{R}^{T\times 3\times H\times W}$, where $C$ is the channel dimension of the VAE latent, and $t, h, w$ are it's compressed spatiotemporal resolutions. A video diffusion model defines a forward noising process:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{1}$$

where $\bar{\alpha}_t$ is determined by the noise schedule. The denoising network is commonly trained to predict either the sampled noise $\boldsymbol{\epsilon}$, the clean latent $\mathbf{x}_0$, or a velocity parameterization derived from the two. The standard formulation assumes that the entries of $\boldsymbol{\epsilon}$ are independent and identically distributed (i.i.d.) standard Gaussian variables. In particular, for two different spatiotemporal latent locations $p$ and $q$,

$$\mathbb{E}[\boldsymbol{\epsilon}_p] = \mathbf{0}, \qquad \mathrm{Cov}(\boldsymbol{\epsilon}_p) = \mathbf{I}_C, \qquad \mathrm{Cov}(\boldsymbol{\epsilon}_p, \boldsymbol{\epsilon}_q) = \mathbf{0} \quad \text{for } p \neq q. \tag{2}$$

This i.i.d. noise assumption is part of the training distribution seen by the pretrained diffusion model. Directly replacing the sampled noise with a strongly locally correlated noise pattern can therefore introduce a train–test mismatch, even when the denoiser is adapted using low-rank adaptation (LoRA).

### 3.2 PRELIMINARY: FEED-FORWARD DYNAMIC GAUSSIAN SPLATTING

Given source video frames $\mathbf{I}^{\text{src}} = \{\mathbf{I}_t^{\text{src}}\}_{t=1}^{T}$, we employ an off-the-shelf feed-forward dynamic Gaussian Splatting model e.g. NoPo4D Balice et al. (2026) to obtain a time-varying 3D representation:

$$\mathcal{G},\, \mathbf{\Pi}_{1:T}^{\text{src}} = \mathcal{F}_{\text{GS}}\left(\mathbf{I}_{1:T}^{\text{src}}\right), \tag{3}$$

where $\mathbf{\Pi}_t^{\text{src}}$ contains the estimated source-camera parameters and

$$\mathcal{G} = \{\mathcal{G}_i\}_{i=1}^{N}, \qquad \mathcal{G}_i = (\bar{\boldsymbol{\mu}}_i,\, \boldsymbol{\mu}_i(t),\, \mathbf{\Sigma}_i(t),\, \alpha_i,\, \mathbf{c}_i), \tag{4}$$

denotes the set of 4D Gaussian primitives. Each primitive has time-varying geometry and appearance together with a time-invariant, view-independent canonical anchor $\bar{\boldsymbol{\mu}}_i \in \mathbb{R}^3$ used to identify it across motion and views. For unobserved regions, we augment the reconstruction with a finite enclosing cubemap. Implementation details of this 4D representation are provided in Appendix A.5. Background samples are obtained by intersecting each target camera ray with this box, which assigns every background pixel an explicit 3D surface point

$$\mathbf{X}_p^{\text{cube}} = \mathbf{o}_v + s_p \mathbf{d}_p, \qquad s_p = \min\{s > 0 : \mathbf{o}_v + s\mathbf{d}_p \in \partial\mathcal{B}\}, \tag{5}$$

where $\mathbf{o}_v$ and $\mathbf{d}_p$ are the target-camera centre and ray direction. Given a target pose $\mathbf{\Pi}_t^{\text{tar}}$, the novel-view RGB frame is obtained by compositing the rasterized Gaussian foreground with this background:

$$\widetilde{\mathbf{I}}_t^{\text{tar}} = \mathbf{C}_t^{\text{GS}} + \left(1 - \mathbf{A}_t^{\text{GS}}\right) \odot \mathbf{C}_t^{\text{cube}}, \tag{6}$$

where $(\mathbf{C}_t^{\text{GS}}, \mathbf{A}_t^{\text{GS}})$ are the rasterized foreground color and opacity. Rendering Eq. 6 over target poses produces the conditioning video used by our diffusion model.

### 3.3 CANONICAL 3D NOISE LATENT

We represent the source video by the set of 4D Gaussian primitives $\mathcal{G} = \{\mathcal{G}_i\}_{i=1}^{N}$ of Eq. 4, each carrying a canonical anchor position $\bar{\boldsymbol{\mu}}_i \in \mathbb{R}^3$ alongside its time-varying Gaussian Splatting attributes. We additionally assign each primitive an independent high-dimensional noise vector

$$\mathbf{z}_i \in \mathbb{R}^H, \qquad \mathbf{z}_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{I}_H), \tag{7}$$

where $H$ denotes the dimension of the canonical 3D noise vectors and is chosen larger than the channel dimension $C$ of the VAE latent features, whose typical values are $C \in \{16, 48\}$. The vectors $\{\mathbf{z}_i\}$ are sampled once and shared across all source and target views. Crucially, $\mathbf{z}_i$ is attached to the identity of primitive $i$ and is a function of neither time nor viewpoint: a dynamic primitive whose center, covariance, or opacity varies over the sequence carries the same $\mathbf{z}_i$ throughout.

Unobserved regions are handled by the same construction: we attach an i.i.d. noise vector $\mathbf{z}_j^{\text{cube}} \in \mathbb{R}^H$ to every texel $j$ of the enclosing box of Sec. 3.2, and a background pixel reads the texel containing its ray–box intersection $\mathbf{X}_p^{\text{cube}}$.

Given the target camera pose at time $t$, rasterizing all primitive noise features, contributed into pixel $p$ of 2D camera plane, gives the noise vector in camera view:

$$\mathbf{Y}_p = \mathcal{R}_{\text{GS}}(\mathcal{G}(t), \mathbf{\Pi}_t^{\text{tar}}) = \sum_{i=1}^{N} w_{p,i}\mathbf{z}_i \in \mathbb{R}^H, \tag{8}$$

where $w_{p,i}$ denotes the contribution of the $i$-th Gaussian point to pixel $p$. The corresponding pixel $q$ in a different view that observes the same scene region shares contributing primitives and therefore depends on the same canonical noise vectors. In particular, their cross-covariance is

$$\text{Cov}(\mathbf{Y}_p, \mathbf{Y}_q) = \left(\sum_{i=1}^{N} w_{p,i} w_{q,i}\right) \mathbf{I}_H. \tag{9}$$

This shared dependence establishes cross-view noise consistency through the underlying 3D correspondence. The geometric correspondence underlying this result is proved for arbitrary camera poses and time-dependent 4D deformations at a fixed timestamp (Appendix A.4). Since $\mathbf{Y}_p$ is a linear combination of independent Gaussian vectors, its marginal distribution and normalized feature

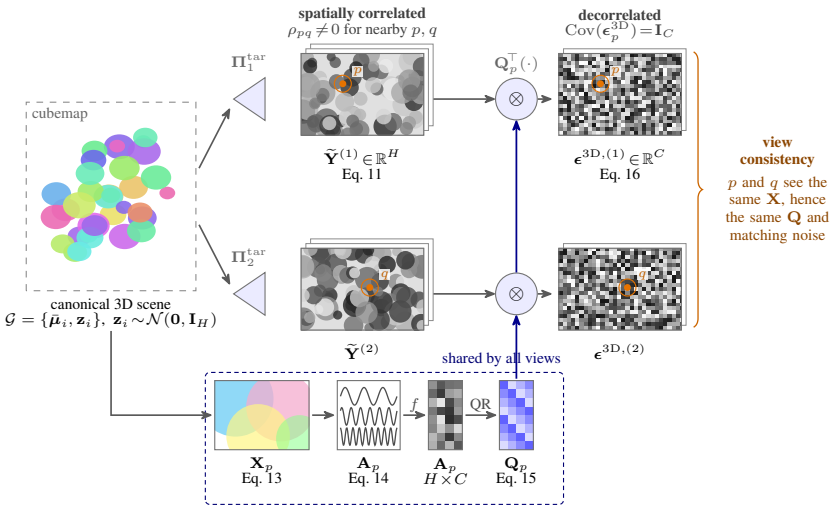


Figure 2: **View-consistent whitening** Noise vectors $\mathbf{z}_i$ are attached to 3D primitives and the cubemap; rasterization therefore provides cross-view correspondence, while overlapping footprints induce local correlation. We reduce this correlation with a per-pixel orthonormal transformation $\mathbf{Q}_p$ derived from the rasterized *canonical anchor* position $\mathbf{X}_p$ using random Fourier features and reduced QR (dashed box). Because $\mathbf{Q}_p$ depends on $\mathbf{X}_p$ rather than screen coordinates, pixels observing the same surface point share the transformation across views and over time. The resulting noise pattern (right) is approximately i.i.d. Gaussian within each frame while remaining cross-view synchronized.

are

$$\mathbf{Y}_p \sim \mathcal{N}\left(\mathbf{0}, v_p \mathbf{I}_H\right), \qquad v_p = \sum_{i=1}^{N} w_{p,i}^2. \tag{10}$$

$$\widetilde{\mathbf{Y}}_p = \frac{\mathbf{Y}_p}{\sqrt{v_p + \varepsilon}} = \frac{\sum_i w_{p,i} \mathbf{z}_i}{\sqrt{\sum_i w_{p,i}^2 + \varepsilon}}, \tag{11}$$

where Eq. 11 normalizes the rasterized feature using the variance factor $v_p$ and $\varepsilon > 0$ provides numerical stability. The $\widetilde{\mathbf{Y}}_p$ has approximately zero mean and identity marginal covariance. Despite this normalization, as shown in Eq. 9, nearby pixels sharing Gaussian primitives remain correlated:

$$\mathrm{Cov}\left(\widetilde{\mathbf{Y}}_p, \widetilde{\mathbf{Y}}_q\right) = \rho_{pq} \mathbf{I}_H, \qquad \rho_{pq} = \frac{\sum_i w_{p,i} w_{q,i}}{\sqrt{(v_p + \varepsilon)(v_q + \varepsilon)}}. \tag{12}$$

While this shared structure provides the desired **cross-view correspondence**, overlapping Gaussian footprints also introduce undesirable local correlations within each rendered noise map. Neighboring pixels typically share several of the same contributing primitives, so their noise values remain far from independent even after the per-pixel normalization above. Because this spatial correlation deviates from the i.i.d. Gaussian distribution assumed by the pretrained diffusion model, it can degrade generation quality when the model is only lightly adapted via fine-tuning (e.g. using LoRA). We address this next with a 3D-aware whitening procedure that reduces local spatial correlations from the rasterized noise while preserving its cross-view and temporal consistency.

### 3.4 View-Consistent Whitening

The rasterized noise from Sec. 3.3 is locally correlated because neighboring pixels may share several Gaussian primitives. We reduce this correlation with a spatially varying projection indexed by canonical 3D position rather than screen coordinates. Consequently, corresponding surface points in different views and at different times use approximately the same projection, while nearby pixels with distinct canonical positions can use different projections.

To obtain the canonical 3D location of pixel $p$, we treat each canonical anchor position $\bar{\boldsymbol{\mu}}_i$ as a rasterized attribute, in the same way that color is rasterized in Eq. 6. This gives the canonical-

coordinate map in screen space:

$$\mathbf{X}_p = \frac{\sum_{i=1}^{N} w_{p,i} \bar{\boldsymbol{\mu}}_i}{\sum_{i=1}^{N} w_{p,i} + \varepsilon} \in \mathbb{R}^3. \tag{13}$$

The weights $w_{p,i}$ come from the deformed geometry at time $t$, so $\mathbf{X}_p$ describes the observed surface in canonical rather than time-varying world space. Using $\boldsymbol{\mu}_i(t)$ would make a moving surface traverse the transformation field and change its whitened noise over time; canonical anchors instead tie $\mathbf{X}_p$ to primitive identity.

For regions outside the source-video coverage that contain no reconstructed primitives, we get $\mathbf{X}_p$ from the ray–box intersection $\mathbf{X}_p^{\text{cube}}$ defined in Eq. 5; corresponding pixels therefore approximately share the same canonical coordinate.

The rasterized coordinate map from Eq. 13 has only three channels, representing 3D positions projected into the camera plane. Since the rasterized noise usually has many more channels than three, we construct a higher-dimensional representation of $\mathbf{X}_p$ using fixed Random Fourier Features (RFF) Tancik et al. (2020). The implementation forms the projection frame directly from fixed frequency vectors and per-output-channel phases:

$$[\mathbf{A}_p]_{a,c} = \sqrt{\frac{2}{H}} \sin\left(2\pi s\, \boldsymbol{\omega}_a^\top \mathbf{X}_p + \phi_{a,c}\right), \qquad \mathbf{A}_p \in \mathbb{R}^{H \times C}, \tag{14}$$

where $\boldsymbol{\omega}_a \in \mathbb{R}^3$ are fixed frequency vectors, $\phi \in [0, 2\pi)^{H \times C}$ contains fixed phases, and $s$ is a scene-dependent frequency scale. The resulting matrix provides multiple fixed sinusoidal responses to the same canonical position. Because the frequencies and phases are view- and time-independent, the resulting representation is consistent across views and over time. We compute the reduced QR factorization:

$$\mathbf{A}_p = \mathbf{Q}_p \mathbf{R}_p, \qquad \mathbf{Q}_p^\top \mathbf{Q}_p = \mathbf{I}_C, \tag{15}$$

and use $\mathbf{Q}_p$ as the local transformation which is applied on rasterized noise $\widetilde{\mathbf{Y}}_p$ (see Figure 2). The frequency scale controls the trade-off between decorrelation and stability: higher frequencies improve local decorrelation but increase sensitivity to reconstruction and rasterization errors. When canonical positions are sufficiently separated relative to the Fourier wavelengths, the high-dimensional feature ($H \gg C$) gives neighboring projection subspaces room to become nearly orthogonal while retaining shared 3D structure. The local-decorrelation behavior of this transformation field is formalized in Appendix A.3, under an idealized assumption that distinct canonical anchors induce generic, approximately independent projection bases. The final structured noise at pixel $p$ is defined as

$$\boldsymbol{\epsilon}_p^{\text{3D}} = \mathbf{Q}_p^\top \widetilde{\mathbf{Y}}_p = \frac{\mathbf{Q}_p^\top \sum_i w_{p,i} \mathbf{z}_i}{\sqrt{\sum_i w_{p,i}^2 + \varepsilon}} \in \mathbb{R}^C. \tag{16}$$

Since $\mathbf{Q}_p$ has orthonormal columns, the marginal covariance is approximately

$$\text{Cov}\left(\boldsymbol{\epsilon}_p^{\text{3D}}\right) = \mathbf{Q}_p^\top \mathbf{Q}_p = \mathbf{I}_C, \tag{17}$$

up to the numerical stabilization and rasterization approximations in Eq. 11. Each projected noise vector therefore has the standard Gaussian marginal expected by the pretrained diffusion model. The remaining cross-pixel covariance is

$$\text{Cov}\left(\boldsymbol{\epsilon}_p^{\text{3D}}, \boldsymbol{\epsilon}_q^{\text{3D}}\right) = \rho_{pq} \mathbf{Q}_p^\top \mathbf{Q}_q. \tag{18}$$

Eq. 18 separates the correlation into the primitive-overlap term $\rho_{pq}$ and the projection term $\mathbf{Q}_p^\top \mathbf{Q}_q$. A varying canonical-position field (and consequently varying $\mathbf{A}_p$) can make the latter small for nearby pixels, whereas a constant or screen-space-dependent field would not provide view-consistent decorrelation. The method is therefore *view-consistent approximate whitening*, not exact covariance whitening: shared primitives still prevent guaranteed independence, but the marginal remains close to the pretrained model's standard Gaussian distribution. The construction provides three properties. *View consistency* follows because corresponding views use approximately the same anchor blend, projection, and mixture of primitive noise vectors. *Temporal consistency* follows because tracked primitives retain their anchors and noise vectors, so the whitened noise follows the object rather than the image plane. The correspondence is exact when the same primitives dominate the rasterization; visibility changes or primitive re-instantiation can still introduce gradual variation in the noise mixture and projection. *Local decorrelation* follows because neighboring pixels generally use different transformations.

### 3.5 SYNCHRONIZED VIDEO GENERATION

We perform synchronized video generation by LoRA fine-tuning a pretrained I2V diffusion model while replacing its i.i.d. Gaussian noise with the view-consistent whitened rasterized noise (Eq. 16). This preserves the pretrained model's standard-Gaussian marginal noise statistics while injecting a shared 3D correspondence signal across views and frames. The video Diffusion Transformer (DiT) uses the Rectified Flow framework Esser et al. (2024), where noising follows straight paths between the clean latent distribution and the structured noise distribution:

$$\begin{aligned} \mathbf{x}_\tau &= (1-\tau)\mathbf{x}_0 + \tau\boldsymbol{\epsilon}^{\mathrm{3D}}, \\ d\mathbf{x}_\tau &= \mathcal{D}_{\theta,\phi}(\mathrm{Concat}_T\left(\mathbf{x}_\tau, \mathbf{x}^c\right), \tau)\, d\tau \end{aligned} \tag{19}$$

Here $\mathcal{D}_{\theta,\phi}$ denotes the DiT with frozen pretrained parameters $\theta$ and learnable LoRA parameters $\phi$. The tensors $\mathbf{x}_0, \boldsymbol{\epsilon}^{\mathrm{3D}} \in \mathbb{R}^{t\times C\times h\times w}$ are the clean target-video VAE latent and its associated rasterized noise, respectively. The conditioning tokens $\mathbf{x}^c$ contain the source-video latent, the rendered RGB-video latent, or both, and are temporally concatenated to the target tokens with extended rotary position encoding. After the DiT blocks, the conditioning tokens are discarded and the output head is applied only to the noised target tokens. Thus, the conditioning stream provides context but is never denoised and never enters the training objective. Only the LoRA parameters $\phi$ are optimized.

## 4 EXPERIMENTS

### 4.1 DATASETS

Because real-world datasets rarely provide synchronized videos of dynamic scenes, we train on synthetic dynamic videos and real-world static videos. For synthetic training, we use 36K video pairs from MultiCamVideo (MCV), introduced by ReCamMaster Bai et al. (2025a), which provides synchronized videos and camera trajectories rendered from dynamic scenes in Unreal Engine 5. To reduce the synthetic-to-real domain gap, we additionally sample 2K DL3DV-10K videos Ling et al. (2024) for real-world training.

DL3DV contains monocular videos of static scenes rather than synchronized multi-view captures, so we construct pseudo-synchronized pairs via temporal folding, following CVD Kuang et al. (2024). For each $2l-1$-frame clip, we split at the center frame $(l+1)$ and reverse the first segment, yielding two segments that share their starting frame and then follow different temporal trajectories. Since the scenes are static, this reversal is physically valid, and each folded pair provides two temporally aligned observations of the same scene along distinct trajectories, with separate associated camera poses. Applying this procedure to the first two buckets of DL3DV yields about 13.5K video pairs.

**Multi-ViewBench.** For evaluation on real-world videos we curate *Multi-ViewBench*, a 200-clip benchmark drawn from the Mixkit subset of Open-Sora-Plan Lin et al. (2024). Because reshooting a real video admits no ground-truth target view, the composition of such a set is easy to bias by hand and hard for others to reproduce, so we select it automatically: a vision-language model judges every candidate clip once against six fixed admissibility gates and four semantic attributes, and every subsequent decision is a deterministic function of those recorded judgements. Admissible clips must be a single continuous real-time shot, landscape, and captured by a physically moving camera; survivors are ranked by a *richness* index rewarding depth layering, occlusion, and non-rigid motion. We report two disjoint splits of 100 clips each, defined by the model's composition judgement: *standard* holds the richest single-subject scenes, and *complex* the richest multi-subject scenes with cluttered geometry, drawn from a category-balanced pool so that no Mixkit category monopolizes the candidates. The complex split is the harder of the two by construction, with a mean richness of 71.2 against 30.9 for standard. We release the code, the per-clip judgements, and the frozen split CSVs; Sec. A.1 documents the pipeline, the per-stage counts, and its limitations in full.

All training and evaluation videos are standardized to 5 sec. clips at 15 fps and a resolution of 480P.

### 4.2 METRICS

We evaluate cross-view synchronization, visual quality, camera accuracy, and perceptual consistency. Synchronization uses high-confidence matched pixels in corresponding regions (Mat. Pix.),

Table 1: **Quantitative comparison on view synchronization, temporal consistency and camera accuracy** on Multi-ViewBench videos. Metrics are defined in Sec. 4.2.

| Method | Generation Length | View Synchronization | | | | Temporal Consistency | Camera Accuracy | |
|---|---|---|---|---|---|---|---|---|
| | | Mat. Pix. ↑ | | FVD-V ↓ | CLIP-V ↑ | CLIP-F ↑ | RotErr ↓ | TransErr ↓ |
| | | Dynamic | Static | | | | | |
| DaS Gu et al. (2025) | 49 | 494 | 1251 | 315.69 | 86.01 | 97.40 | 0.1549 | 0.2164 |
| Go-with-the-Flow Burgert et al. (2025) | 49 | 687 | 1277 | 296.30 | 87.44 | 96.78 | 0.1225 | 0.1109 |
| ReCamMaster Bai et al. (2025a) | 75 | 1073 | 1428 | 233.70 | 88.06 | 97.39 | 0.1665 | 0.1778 |
| ArtiFixer-14B De Lutio et al. (2026) | 75 | 1128 | 1937 | 195.55 | 88.99 | 97.55 | **0.0780** | **0.0529** |
| Reshoot-Anything Paliwal et al. (2026) | 69 | 1268 | **1989** | 202.22 | 89.96 | 97.82 | 0.1120 | 0.0838 |
| **SyncVid (Ours)** | **75** | **1332** | 1834 | **146.99** | **91.66** | **97.87** | 0.151 | 0.121 |

Table 2: **Quantitative comparison on VBench Huang et al. (2024) metrics**.

| Method | Aesthetic Quality ↑ | Imaging Quality ↑ | Temporal Flickering ↑ | Motion Smoothness ↑ | Subject Consistency ↑ | Background Consistency ↑ |
|---|---|---|---|---|---|---|
| DaS Gu et al. (2025) | 49.15 | 50.34 | 96.70 | 98.54 | 84.74 | 90.81 |
| Go-with-the-Flow Burgert et al. (2025) | 49.91 | 51.98 | 95.63 | 98.00 | 84.87 | 90.69 |
| ReCamMaster Bai et al. (2025a) | 48.12 | 49.51 | **97.22** | **98.89** | 85.41 | 89.91 |
| ArtiFixer De Lutio et al. (2026) | 48.10 | 52.61 | 96.54 | 98.81 | 85.67 | 90.41 |
| Reshoot-Anything Paliwal et al. (2026) | 47.85 | 51.72 | 96.16 | 98.45 | 85.94 | **92.01** |
| **SyncVid (Ours)** | **52.07** | **56.03** | 96.45 | 98.71 | **88.66** | 91.24 |

Table 3: **Human preference evaluations.** Percentage of blind pairwise comparisons.

| SyncVid (Ours) | Tie | ArtiFixer De Lutio et al. (2026) | SyncVid (Ours) | Tie | Reshoot-Any. Paliwal et al. (2026) |
|---|---|---|---|---|---|
| **78**% | 15% | 7% | **47**% | 31% | 22% |

cross-view FVD (FVD-V), and cross-view CLIP similarity (CLIP-V). Temporal consistency uses frame consistency (CLIP-F); following ReCamMaster Bai et al. (2025a), camera accuracy is measured by rotation and translation errors (RotErr/TransErr) between the conditioning and recovered trajectories. We also report six no-reference VBench Huang et al. (2024) scores for aesthetic and imaging quality, temporal flickering and motion smoothness, and subject/background consistency. Finally, a blind pairwise study presents the source and two methods' renderings of the same target trajectory in randomized left/right order; participants select the better result or no preference, covering 50 / 200 comparisons on two disjoint sets of 25 **Multi-ViewBench** scenes (SyncVid vs. Artifixer and SyncVid vs. Reshoot-Anything).

### 4.3 Main results

Following the evaluation protocol of ReCamMaster Bai et al. (2025a), we compare SyncVid with recent state-of-the-art methods on the real-world Multi-ViewBench test set. Table 1 shows that our method provides the strongest overall cross-view synchronization. It achieves the highest number of matched pixels in dynamic regions (1,332) and the second-highest number in static regions (1,834), behind Reshoot-Anything. It also obtains the lowest cross-view FVD (146.99) and the highest cross-view CLIP similarity (91.66), indicating that its generated views are both perceptually aligned and visually similar. In addition, it achieves the highest temporal-consistency score (CLIP-F = 97.87). Camera accuracy is not its main advantage: ArtiFixer obtains the lowest rotation and translation errors, while our method remains competitive.

The VBench results in Table 2 show a similar advantage in perceptual quality. Our method achieves the highest aesthetic quality (52.07), imaging quality (56.03), and subject consistency (88.66). Its temporal-flickering and motion-smoothness scores are close to the best reported values, and its background consistency (91.24) is second only to Reshoot-Anything. Overall, these results suggest that the proposed view-consistent noise improves cross-view alignment and content fidelity without substantially compromising temporal or background stability. Table 3 shows the user study results.

### 4.4 Ablations

We ablate the main design choices of our method on a held-out set of OpenSora videos, using the lighter Wan2.2-5B backbone to keep the study computationally affordable. Results are reported in Table 4. Whitening improves all four reported metrics over both standard noise and unwhitened




Figure 3: **Qualitative comparison** on a Multi-ViewBench example with an Arc-right trajectory.

Table 4: **Ablations.** Visual-quality and view-synchronization metrics on various factors.

| Variant | | View Synchronization | | | Temporal Consistency |
|---|---|---|---|---|---|
| | | Mat. Pix. ↑ | FVD-V ↓ | CLIP-V ↑ | CLIP-F ↑ |
| Noise | standard | 222 | 518.77 | 88.76 | 97.34 |
| | w/o whitening | 208 | 468.39 | 88.53 | 96.92 |
| | w/ whitening | 267 | 461.44 | 91.05 | 97.61 |
| Conditioning | +none (I2V) | 224 | 830.00 | 87.53 | 97.80 |
| | +render view | 231 | 574.50 | 89.34 | 97.59 |
| | +source view | 319 | 530.42 | 91.41 | 97.95 |
| | +both | 267 | 461.44 | 91.05 | 97.61 |
| Fine tuning | LoRA 128 | 214 | 494.43 | 88.33 | 96.97 |
| | LoRA 256 | 267 | 461.44 | 91.05 | 97.61 |
| | LoRA 512 | 206 | 490.21 | 88.38 | 96.92 |
| Backbone | Wan2.1-1.3B | 217 | 689.30 | 88.23 | 97.25 |
| | Wan2.2-5B | 267 | 461.44 | 91.05 | 97.61 |
| | Wan2.1-14B | 308 | 442.20 | 90.88 | 97.49 |
| | Wan2.2-14B | 313 | 438.69 | 91.17 | 97.75 |

noise: it increases Mat. Pix. from 222 to 267, reduces FVD-V from 518.77 to 461.44, and raises CLIP-V from 88.76 to 91.05, while slightly improving CLIP-F. The conditioning ablation reveals a complementary trade-off. Adding the source view yields the strongest direct synchronization, with the highest Mat. Pix. (319), CLIP-V (91.41), and CLIP-F (97.95). Using both the rendered and source views instead gives the lowest FVD-V (461.44), suggesting that the additional rendered view improves distributional visual similarity even though it does not maximize pixel-level matching. We favor this configuration because FVD-V, which compares the distributions of generated and reference views, better reflects the overall quality of the generated videos than pixel-level matching.

LoRA 256 is the best rank on all four metrics, and the larger DiT improves almost every metric. Our final model therefore uses whitening, joint source- and rendered-view conditioning, LoRA 256, and the larger backbone.

## 5 Conclusion

We presented SyncVid, a framework for synchronized novel-view video generation that coordinates diffusion at the source of its stochasticity. By attaching persistent noise to a canonical 4D representation and rasterizing it into target cameras, SyncVid turns scene correspondence into a property of the diffusion prior rather than a responsibility of image-plane warping or additional cross-view architecture. A view-consistent whitening step keeps this structured prior compatible with the distribution expected by the pretrained model, allowing the approach to be integrated through lightweight adaptation without modifying the diffusion backbone. These findings suggest that synchronized generation can benefit from treating noise as a world-level representation: the same stochastic ingredients can follow content across viewpoints and time while leaving the model free to synthesize newly revealed regions.

## AI USE STATEMENT

**This section is *required* and does not count toward the page limit.**

We used AI assistants for writing and presentation: proofreading, grammar checking and preparing the supplementary HTML webpage. The core contribution was implemented entirely by hand, validated against a suite of developer tests, and peer-reviewed by the authors; no AI system was involved in writing that code. All results, figures, and claims reported in this paper were verified by the authors.

## REFERENCES

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 *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 14834–14844. IEEE, 2025a.

Jianhong Bai, Menghan Xia, Xintao Wang, Ziyang Yuan, Zuozhu Liu, Haoji Hu, Pengfei Wan, and Di Zhang. Syncammaster: Synchronizing multi-camera video generation from diverse viewpoints. In *International Conference on Learning Representations*, volume 2025, pp. 58038–58060, 2025b.

Matteo Balice, Yanik Kunzi, Chenyangguang Zhang, Matteo Matteucci, Marc Pollefeys, and Sungwhan Hong. No pose, no problem in 4d: Feed-forward dynamic gaussians from unposed multi-view videos. *arXiv preprint arXiv:2605.22190*, 2026.

Ryan Burgert, Yuancheng Xu, Wenqi Xian, Oliver Pilarski, Pascal Clausen, Mingming He, Li Ma, Yitong Deng, Lingxiao Li, Mohsen Mousavi, et al. Go-with-the-flow: Motion-controllable video diffusion models using real-time warped noise. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13–23. IEEE, 2025.

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris Coll-Vinent, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. Sam 3: Segment anything with concepts. In *International conference on learning representations*, volume 2026, pp. 138846–138923, 2026.

Pascal Chang, Jingwei Tang, Markus Gross, and Vinicius Da Costa De Azevedo. How i warped your noise: a temporally-correlated noise prior for diffusion models. In *International Conference on Learning Representations*, volume 2024, pp. 30015–30046, 2024.

Riccardo De Lutio, Tobias Fischer, Yen-Yu Chang, Yuxuan Zhang, Zhangjie Wu, Xuanchi Ren, Tianchang Shen, Katarína Tóthová, Zan Gojcic, and Haithem Turki. Artifixer: Enhancing and extending 3d reconstruction with auto-regressive diffusion models. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pp. 1–12, 2026.

J Edstedt, D Nordström, Y Zhang, G Bökman, J Astermark, V Larsson, A Heyden, F Kahl, M Wadenbäck, and M Felsberg. Roma v2: Harder better faster denser feature matching. arxiv 2025. *arXiv preprint arXiv:2511.15706*.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

Songwei Ge, Seungjun Nah, Guilin Liu, Tyler Poon, Andrew Tao, Bryan Catanzaro, David Jacobs, Jia-Bin Huang, Ming-Yu Liu, and Yogesh Balaji. Preserve your own correlation: A noise prior for video diffusion models. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22873–22884. IEEE, 2023.

Zekai Gu, Rui Yan, Jiahao Lu, Peng Li, Zhiyang Dou, Chenyang Si, Zhen Dong, Qifeng Liu, Cheng Lin, Ziwei Liu, et al. Diffusion as shader: 3d-aware video diffusion for versatile video generation

control. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pp. 1–12, 2025.

Richard Hartley, Andrew Zisserman, et al. *Multiple view geometry in computer vision*, volume 2. Cambridge university press Cambridge, 2003.

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, 2025.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. Vbench: Comprehensive benchmark suite for video generative models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21807–21818. IEEE, 2024.

Zhengfei Kuang, Shengqu Cai, Hao He, Yinghao Xu, Hongsheng Li, Leonidas J Guibas, and Gordon Wetzstein. Collaborative video diffusion: Consistent multi-video generation with camera control. *Advances in Neural Information Processing Systems*, 37:16240–16271, 2024.

Bin Lin, Yunyang Ge, Xinhua Cheng, Zongjian Li, Bin Zhu, Shaodong Wang, Xianyi He, Yang Ye, Shenghai Yuan, Liuhan Chen, et al. Open-sora plan: Open-source large video generation model. *arXiv preprint arXiv:2412.00131*, 2024.

Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, et al. Dl3dv-10k: A large-scale scene dataset for deep learning-based 3d vision. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22160–22169. IEEE, 2024.

Chao Liu and Arash Vahdat. On equivariance and fast sampling in video diffusion models trained with warped noise. *arXiv preprint arXiv:2504.09789*, 2025.

Avinash Paliwal, Adithya Iyer, Shivin Yadav, Muhammad Afridi, and Midhun Harikumar. Reshoot-anything: A self-supervised model for in-the-wild video reshooting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11596–11606, 2026.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20, 2007.

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in neural information processing systems*, 33:7537–7547, 2020.

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, 2024.

Dejia Xu, Yifan Jiang, Chen Huang, Liangchen Song, Thorsten Gernoth, Liangliang Cao, Zhangyang Wang, and Hao Tang. Cavia: Camera-controllable multi-view video diffusion with view-integrated attention. *arXiv preprint arXiv:2410.10774*, 2024a.

Dejia Xu, Weili Nie, Chao Liu, Sifei Liu, Jan Kautz, Zhangyang Wang, and Arash Vahdat. Camco: Camera-controllable 3d-consistent image-to-video generation. *arXiv preprint arXiv:2406.02509*, 2024b.

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.

Farhad G Zanjani, Hong Cai, and Amirhossein Habibian. Enhancing novel view synthesis via geometry grounded set diffusion. *arXiv preprint arXiv:2601.07540*, 2026.

# A SUPPLEMENTARY

## A.1 AUTOMATED EVALUATION SET CURATION FOR OPENSORA

Reshooting a real video admits no ground-truth target view, so the only thing a real-world benchmark can measure is behavior on the clips it happens to contain. That makes the composition of the test set part of the result. A set assembled by hand can flatter a method through nothing more than which scenes its author found compelling, and it cannot be reproduced by anyone else. We therefore built Multi-ViewBench, the 200-clip evaluation set used in Sec. 4.3, so that every accept or reject decision is either a recorded model judgement or a deterministic function of recorded judgements. No clip was ever chosen, dropped, or reordered by inspection.

The pipeline has two stages. Stage 1 asks a vision-language model to judge each candidate clip once and writes the judgement to disk. Stage 2 never calls a model again: it filters, de-duplicates, and ranks using only the fields Stage 1 already recorded, so the entire path from corpus to final CSV can be replayed offline from the released judgements.

### A.1.1 SOURCE CORPUS

Candidates come from the Mixkit subset of Open-Sora-Plan v1.1.0 Lin et al. (2024), which contains 9,194 stock-footage `.mp4` clips filed under 40 topical category folders (`forest`, `Street`, `Couple`, `Bicycle`, and so on). Two properties of this corpus shape the pipeline. First, the category folders are editorial tags rather than a partition: the same physical clip is frequently filed under several categories, so the corpus contains substantially fewer distinct videos than files. Second, the clips are heterogeneous in exactly the ways that matter for camera-controlled generation, mixing locked-off tripod shots, time-lapses, portrait-orientation footage, and heavy optical effects with the moving-camera material we actually want.

### A.1.2 STAGE 1: STRUCTURED VISION-LANGUAGE JUDGEMENT

Each clip is judged independently by Gemini 3.1 Pro (`gemini-3.1-pro-preview`), accessed through an OpenAI-compatible inference gateway. We sample 12 uniformly spaced frames per clip, rescale each so its longest side is at most 512 pixels, JPEG-encode them, and submit the sequence with a fixed system prompt at temperature 0.2. The response is constrained to a JSON schema, so the model cannot answer in prose or omit a field, and every reply is machine-readable without parsing heuristics. One JSON file is written per clip, which makes the pass resumable: an interrupted run skips clips whose output already exists.

The schema, summarized in Table 5, has four parts. Six boolean *gates* encode hard admissibility requirements. A *composition bucket* assigns the clip to `simple` or `complex` and is the field that later defines our two splits: `simple` denotes a single subject under smooth tracking against a clean background, and `complex` denotes multiple subjects, cluttered environments, distinct depth layers, or dynamic backgrounds. Four *semantic scores* rate the properties that make a clip informative for view synthesis. Finally, three free-text *reasoning* fields record why the model answered as it did; these are released, and Stage 2 consults them in exactly one narrow place (Sec. A.1.4).

### A.1.3 THE RICHNESS INDEX

Gates decide admissibility, but among admissible clips we still need an ordering, and we want it to prefer clips that actually exercise synchronized generation. A clip is informative to the degree that it has depth to be parallaxed, geometry that occludes and disoccludes as the viewpoint moves, a camera that moves at all, and subjects that deform rather than translate rigidly. We combine the four semantic scores into a single *richness index*

$$\rho = \underbrace{g}_{\text{parallax}} \cdot \underbrace{o}_{\text{occlusion}} \cdot w_{\text{cam}} \cdot w_{\text{nr}}, \tag{20}$$

where $g, o \in \{1, \ldots, 5\}$ are `geometric_parallax` and `occlusion_potential`, and the two weights are read off the categorical labels: $w_{\text{cam}} = 2.0$ for `dynamic`, 1.2 for any optical-effect label, and 1.0 for `static`; $w_{\text{nr}} = 1.0, 1.2, 1.5, 2.0$ for `none`, `low`, `mid`, `high` respectively. Richness ranges over $[1, 100]$.

Table 5: **Stage-1 judgement schema.** The model must return every field. Gates are hard requirements: a clip is admissible only if all six are true. The composition bucket defines the *standard/complex* split, and the semantic scores feed the richness index of Sec. A.1.3.

| Field | Range | Requirement or meaning |
|---|---|---|
| *Gates* (all must be true) | | |
| `is_single_continuous_shot` | bool | No cuts or transitions. |
| `is_real_time_playback` | bool | No time-lapse, hyper-lapse, or slow motion. |
| `is_physically_plausible` | bool | No VFX blends, text overlays, or screen-in-screen. |
| `is_motion_blur_acceptable` | bool | No motion blur severe enough to destroy structure. |
| `is_focal_blur_acceptable` | bool | Not excessively out of focus. |
| `is_depth_of_field_acceptable` | bool | No extreme shallow depth of field. |
| *Composition* | | |
| `composition_bucket` | 2 labels | `simple` for a single tracked subject on a clean background, `complex` for multiple subjects, clutter, or depth layers. |
| *Semantic scores* | | |
| `geometric_parallax` | 1–5 | 1 is flat, 5 is deep multi-layered depth. |
| `occlusion_potential` | 1–5 | 1 is no meaningful occlusion, 5 is heavy self-occlusion or disocclusion. |
| `camera_motion` | 5 labels | `static`, `dynamic`, or one of three optical-effect labels. |
| `non_rigid_motion` | 4 labels | `none`, `low`, `mid`, or `high` object articulation. |
| *Reasoning* (released, used only as described in Sec. A.1.4) | | |
| `gates_reasoning` | text | Justification for the six gate verdicts. |
| `composition_reasoning` | text | Justification for the composition bucket. |
| `scores_reasoning` | text | Justification for the four semantic scores. |

The product form is deliberate. Depth without occlusion, or occlusion without camera motion, describes a clip that is easy for a single-view model and therefore uninformative here, and a product refuses to compensate for a low factor with a high one. Because Stage 2 admits only moving-camera clips, $w_{\mathrm{cam}} = 2.0$ holds throughout the final pool and richness is effectively $2go\, w_{\mathrm{nr}}$ there.

### A.1.4 Stage 2: deterministic filtering

Stage 2 reduces the scored corpus to a candidate pool through five filters applied in a fixed order. Table 6 reports the count surviving each.

**Gate pass.** A clip is kept only if all six gates are true.

**Landscape.** We probe each survivor with a video decoder and keep clips whose frame width exceeds their height, discarding portrait and square footage that no baseline is trained to reshoot. This is the only filter that reads pixels; frame dimensions are cached so reruns do not repeat it.

**De-duplication.** Because one clip may be filed under several categories, Stage 1 scored each copy independently, and copies routinely disagree with one another. We collapse to one row per unique file name, keeping the highest-richness copy and breaking ties deterministically on (category, judgement filename) so the outcome does not depend on directory iteration order. The discarded copies' categories are preserved in an `alias_categories` field rather than thrown away.

**Static-source removal.** A clip whose camera body never moves gives a camera-controlled model no viewpoint baseline to reshoot from, so we keep only physically moving cameras. The `camera_motion` label carries this directly for `static` and `dynamic`. The optical-effect labels are genuinely ambiguous, since a rack focus or lens flare is usually captured on a locked-off tripod but occasionally accompanies a real dolly move. This is the one place where the reasoning text is consulted: a regular expression looks for explicit camera-body motion (pan, tilt, track, dolly, handheld) to adjudicate the optical-effect clips, and to demote the rare `dynamic` clip whose own justification states the camera is stationary. The text is never used to overrule an unambiguous label.

Table 6: **Curation funnel.** Counts at the time the benchmark was frozen, when 7,561 of the 9,194 corpus clips had been scored. De-duplication runs after the gate and landscape filters but before static-source classification; see Sec. A.1.5 for why the order matters.

| # | Stage | Surviving | Removed |
|---|---|---|---|
| | Stage-1 judgements available | 7,561 | |
| 1 | Gate pass (all six gates true) | 6,617 | 944 gate failures |
| 2 | Landscape (width $>$ height) | 5,973 | 644 portrait or square |
| 3 | De-duplication (unique clips) | 4,541 | 1,432 copies of 1,197 clips |
| 4 | Moving camera body | 2,231 | 2,310 static-source |
| 5 | Optical-effect removal | **2,227** | 4 optical-effect |
| | of which `complex` | 1,823 | |
| | of which `simple` | 404 | |

**Optical-effect removal.** Any clip still carrying a `focal_effects`, `aperture_effects`, or `sensor_effects` label is dropped outright, since lens-driven apparent motion is not viewpoint change.

### A.1.5 Why the filter order matters

Two orderings in Table 6 are load-bearing rather than incidental.

De-duplication runs *after* the gates so that a copy which fails a gate can never win the richness comparison and readmit a clip that should have been rejected. It runs *before* static-source classification so that each physical clip receives exactly one verdict. Were the order reversed, a clip could enter the pool through whichever of its category copies happened to be scored most favorably, which would let editorial filing decisions leak into the benchmark.

Static-source removal precedes optical-effect removal because the two overlap: the adjudication step already reclassifies most optical-effect clips as static, so only 4 clips survive to be dropped by the explicit optical filter. Reversing the two would discard the handful of optical-effect clips that do accompany real camera motion.

### A.1.6 Constructing the two splits

The pool is strongly unbalanced, holding 1,823 complex clips against 404 simple ones, because static-source removal strips proportionally more single-subject tracking shots. The two splits are therefore built differently, and deliberately so.

**Standard (100 clips).** Simple compositions are the comparatively solved case, so this split is a compact reference rather than half the benchmark, and its specification is exactly *take the 100 highest-richness simple clips*. No category cap and no diversity heuristic apply. With only 404 candidates, imposing balance here would trade away richness for very little diversity.

**Complex (100 clips).** A pure richness sort over 1,823 complex clips collapses onto whichever categories are largest, and the supply is severely skewed (150 clips for `forest` against 1 for `Cats`). We therefore construct this split in two steps. First we select a category-balanced set of 1,000 clips in which balance is a hard constraint and richness is maximized only within it. Because an equal quota of $1000/40 = 25$ per category is unavailable from most categories, we allocate quotas by max-min fair water-filling: give every category the same quota, let categories that cannot meet it contribute everything they have, and redistribute the shortfall equally among categories that still have supply, repeating until the budget is spent. This maximizes the smallest category's share, then the next smallest, and degenerates to an equal split when supply permits. Each category then contributes its own richest clips up to its quota, so balance is never traded for richness. The resulting 1,000 clips span all 40 categories with between 1 and 38 clips each, a mean of 25.0, and a normalized category entropy of 0.940 against a maximum of 1. Second, we take the 100 highest-richness clips from this balanced set to form the released split.

The two-step construction is worth stating plainly, because balance and richness pull against each other and the second step partially undoes the first. Balancing at 1,000 determines *which* clips are available to be ranked, preventing the fat categories from monopolizing the candidate set; truncating

Table 7: **The two released splits of Multi-ViewBench.** Both hold 100 clips, are disjoint, and are drawn from the 2,227-clip pool of Table 6. Richness $\rho$ is defined in Eq. 20. Attribute rows give the distribution over the 100 clips.

| | Standard | Complex |
|---|---|---|
| Composition bucket | `simple` | `complex` |
| Candidates in pool | 404 | 1,823 |
| Selection | top 100 by $\rho$ | balance to 1,000, then top 100 by $\rho$ |
| $\rho$ range | 24.0–64.0 | 64.0–100.0 |
| $\rho$ mean | 30.9 | 71.2 |
| $\rho$ median | 32.0 | 64.0 |
| Categories represented | 19 | 19 |
| Largest category | `Music` (24) | `Couple` (26) |
| Clips with alias categories | 20 | 47 |
| Geometric parallax | 2: 57 3: 36 4: 7 | 4: 83 5: 17 |
| Occlusion potential | 2: 9 3: 63 4: 28 | 4: 73 5: 27 |
| Non-rigid motion | low: 1 mid: 4 high: 95 | high: 100 |
| Camera motion | `dynamic`: 100 | `dynamic`: 100 |

to the richest 100 then concentrates on 19 of the 40 categories. What balance buys is that the final 100 is drawn from a category-fair pool rather than from a raw richness sort over the whole corpus, not that the final 100 is itself uniform over categories.

Both selections sort by richness descending with ties broken deterministically on (category, file name), and in both cases the cut falls inside a tie group: the standard split cuts within the $\rho = 24$ tier, and the complex split takes 39 clips above $\rho = 64$ and then 61 of the 210 clips at exactly $\rho = 64$. The tie-break rule is what makes these selections reproducible, and it is the reason the released CSVs are byte-identical across reruns.

### A.1.7 Resulting statistics

Table 7 characterizes the two released splits. They are disjoint by construction, sharing no clip, and together form the 200 clips of Multi-ViewBench. The richness distributions are almost separated: every complex clip scores at least 64, a value that exactly one standard clip attains and none exceeds, against a standard mean of 30.9. The gap is visible in the underlying attributes rather than being an artifact of the index, since every complex clip carries high non-rigid motion and a parallax rating of at least 4, whereas the standard split is concentrated at parallax 2 to 3.

All 200 clips are natively $1920 \times 1080$, and are standardized to 5-second clips at 15 fps and 480p for evaluation as described in Sec. 4.1.

### A.1.8 Reproducibility and release

We release the complete curation trail, not only the final clip list, so that the benchmark can be audited and extended rather than merely consumed:

- the Stage-1 and Stage-2 code, including the verbatim system prompt, the response schema, and the richness weights;
- the per-clip Stage-1 judgement JSONs, covering all gates, scores, and reasoning texts, for every clip scored, including those the pipeline rejected;
- the intermediate pool CSV and both released split CSVs, each carrying all 18 columns, so a reader can recompute any count in Table 6 or reproduce either selection; and
- the exact clip identities and source paths, together with a SHA-256 manifest over the frozen CSVs.

The frozen CSVs are the integrity anchor for every number reported in Sec. 4.3: the splits were fixed before any baseline was evaluated and were not revised afterwards. Stage 2 is deterministic and seed-free, so given the released judgements the splits are recoverable exactly.

### A.2 Novel-Trajectory Synthesis

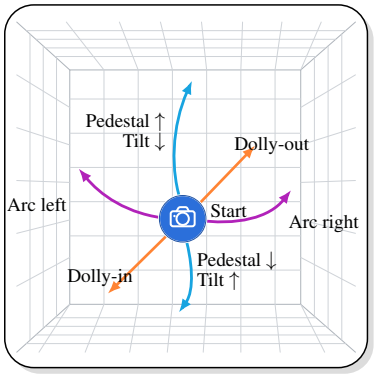


Figure 4: **Schematic of the six synthesized evaluation trajectories**; Every trajectory is a camera-local SE(3) delta $\Delta_t$ composed *after* the source camera's pose at each frame (Eq. 21), so all six depart from the same source pose (`Start`) at $t = 0$ and diverge from it over the clip. Arc left/right (violet) sweep the camera around a pivot ahead of it in the local horizontal (right–forward) plane; Dolly in/out (orange) dolly along the local forward axis toward/away from that pivot; Pedestal-up-Tilt-down and Pedestal-down-Tilt-up (blue) shift the camera along its local down axis while tilting about its local right axis so the same pivot stays centred in frame.

Because each OpenSora scene provides a single source camera, there is no held-out ground-truth viewpoint against which to evaluate reshoots on this data; we instead procedurally synthesize novel target trajectories $\mathbf{\Pi}^{\text{tar}}_{1:T}$ that follow the source camera's own motion rather than being anchored to its initial pose. Concretely, at every frame $t$ we compose a time-varying SE(3) offset $\Delta_t$ *after* the source pose, expressed in the source camera's own local axes (OpenCV convention):

$$\mathbf{\Pi}^{\text{tar}}_t = \mathbf{\Pi}^{\text{src}}_t \, \Delta_t. \tag{21}$$

Since OpenSora sources are handheld or drone footage whose camera keeps moving throughout the clip, anchoring $\Delta_t$ to a fixed world pose would let the novel and source views drift apart in content over the clip; composing it on the current source pose instead keeps the novel view co-moving with the source, offset from it by a structured, time-varying amount. Every $\Delta_t$ is the identity at $t = 0$, so each synthesized trajectory has aligned initial pose with the source trajectory and diverges from it over the clip. We synthesize six trajectory families this way, illustrated schematically in Fig. 4:

**Arc left/right** sweep $\Delta_t$'s rotation up to $60°$ about the source camera's local down axis, pivoting around a point a distance $d$ ahead of the camera.

**Dolly in/out** shift $\Delta_t$ purely along the local forward axis, so the same pivot ends up three times closer or three times farther by the last frame.

**Pedestal-up-Tilt-down** and **Pedestal-down-Tilt-up** translate $\Delta_t$ along the local down axis by up to 30% of the estimated distance to the surface being approached (e.g. the floor), while simultaneously tilting about the local right axis so the same pivot stays centred in frame throughout.

The pivot distance $d$ and the vertical hit-surface distance used to size the pedestal trajectories are both estimated per scene from the reconstructed Gaussians $\mathcal{G}$ (Sec. 3.2). For arc/dolly, $d$ is the depth of the nearest surface in a small patch at the centre of frame 0; for pedestal trajectories, the hit-surface distance is the mean depth over the bottom band of frame 0. Both fall back to the median distance of the scene (based on $\{\bar{\boldsymbol{\mu}}_i\}$) from the source camera when a per-pixel depth estimate is unavailable for example a video from a landscape, so trajectory magnitudes remain sensible regardless of the arbitrary metric scale a feed-forward backbone assigns to a given scene.

### A.3 Local Decorrelation of the Whitening Projection

Sec. 3.4 argues informally that the random-Fourier-feature (RFF) projection field $\mathbf{Q}_p$ (Eq. 15) suppresses the cross-pixel term $\mathbf{Q}_p^\top \mathbf{Q}_q$ in the projected covariance (Eq. 18) whenever two pixels' canonical anchors $\mathbf{X}_p, \mathbf{X}_q$ are distinct. We make this precise with a short argument: two *distinct* 3D anchors are mapped, through the many-harmonic the fixed sinusoidal map of Eq. 14 produces essentially unrelated pre-projection matrices $\mathbf{A}_p, \mathbf{A}_q$; QR factorizing two unrelated matrices yields two unrelated orthonormal bases $\mathbf{Q}_p, \mathbf{Q}_q$ of $\mathbb{R}^H$; and two unrelated $C$-dimensional bases of a much larger $H$-dimensional space are nearly orthogonal to each other, i.e. $\mathbf{Q}_p^\top \mathbf{Q}_q \approx \mathbf{0}$. This is what should suppress the correlation $\rho_{pq}$ in Eq. 18. Note that the target here is the *zero* matrix, not the identity: $\mathbf{Q}_p^\top \mathbf{Q}_p = \mathbf{I}_C$ holds *exactly* by the reduced QR factorization in Eq. 15; it is only the $p \neq q$ case that needs an argument. Together, an exact identity on the diagonal and a near-zero off-diagonal

term is what makes the noise field as a whole resemble an i.i.d. (identity-covariance) Gaussian field, which is the property the pretrained diffusion model expects.

**Assumption 1** (Distinct anchors encode as generic bases)**.** *When $\mathbf{X}_p \neq \mathbf{X}_q$, the direct Fourier map of Eq. 14 uses $H$ frequency vectors $\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_H$ spanning many distinct, incommensurate harmonics (cf. random Fourier features Rahimi & Recht (2007)), so that the resulting $\mathbf{A}_p, \mathbf{A}_q \in \mathbb{R}^{H\times C}$ – carry no shared structure beyond what is forced by $\mathbf{X}_p$ being close to $\mathbf{X}_q$. We formalize "no shared structure" by treating the orthonormal bases $\mathbf{Q}_p, \mathbf{Q}_q$ obtained by QR-factorizing $\mathbf{A}_p, \mathbf{A}_q$ (Eq. 15) as two independent, uniformly random $C$-dimensional subspaces of $\mathbb{R}^H$ (i.e. Haar-random on the Grassmannian) whenever $\mathbf{X}_p \neq \mathbf{X}_q$. This is the sense in which "many harmonics" matters: it is what keeps two different points from being accidentally mapped to correlated bases.*

**Lemma 1** (Random bases in high dimension are nearly orthogonal)**.** *Let $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{H\times C}$ have orthonormal columns, with $\mathbf{V}$ Haar-random on the Grassmannian and independent of the fixed $\mathbf{U}$. Then*

$$\mathbb{E}\big[\mathbf{U}^\top \mathbf{V}\big] = \mathbf{0}, \qquad \mathbb{E}\big[\|\mathbf{U}^\top \mathbf{V}\|_F^2\big] = \frac{C^2}{H}, \tag{22}$$

*so, by Jensen's inequality, $\mathbb{E}\big[\|\mathbf{U}^\top \mathbf{V}\|_F\big] \leq C/\sqrt{H}$, or equivalently $\mathbb{E}\big[\frac{1}{\sqrt{C}}\|\mathbf{U}^\top \mathbf{V}\|_F\big] \leq \sqrt{C/H} \to 0$ as $H \to \infty$ for fixed $C$.*

*Proof.* Let $\mathbf{u}_i, \mathbf{v}_j$ denote the unit-norm columns of $\mathbf{U}, \mathbf{V}$. Since $\mathbf{v}_j$ is a uniformly random unit vector in $\mathbb{R}^H$ independent of the fixed $\mathbf{u}_i$, rotational symmetry gives $\mathbb{E}[\mathbf{u}_i^\top \mathbf{v}_j] = 0$ and, since $\sum_{h=1}^H \mathbb{E}[v_{j,h}^2] = 1$ with every coordinate direction equivalent by symmetry, $\mathbb{E}\big[(\mathbf{u}_i^\top \mathbf{v}_j)^2\big] = \frac{1}{H}$ for every unit vector $\mathbf{u}_i$. Summing over the $C^2$ entries of $\mathbf{U}^\top \mathbf{V}$ gives Eq. 22. This is exact: each column $\mathbf{v}_j$ is marginally uniform on the unit sphere, so the orthogonality among the columns of $\mathbf{V}$ does not affect the second moment. The bound on $\mathbb{E}\|\mathbf{U}^\top \mathbf{V}\|_F$ then follows from $\mathbb{E}[X] \leq \sqrt{\mathbb{E}[X^2]}$. □

**Proposition 1** (Local decorrelation of the whitening projection)**.** *Under Assumption 1, for two pixels $p \neq q$ with distinct anchors $\mathbf{X}_p \neq \mathbf{X}_q$,*

$$\mathbb{E}\big[\mathbf{Q}_p^\top \mathbf{Q}_q\big] = \mathbf{0}, \qquad \frac{1}{\sqrt{C}}\,\mathbb{E}\big[\|\mathbf{Q}_p^\top \mathbf{Q}_q\|_F\big] \leq \sqrt{\frac{C}{H}}, \tag{23}$$

*while $\mathbf{Q}_p^\top \mathbf{Q}_p = \mathbf{I}_C$ exactly, for which the same normalized norm equals 1. Consequently, in the projected cross-covariance of Eq. 18,*

$$\frac{1}{\sqrt{C}}\,\mathbb{E}\big[\|\operatorname{Cov}(\boldsymbol{\epsilon}_p^{\mathrm{3D}}, \boldsymbol{\epsilon}_q^{\mathrm{3D}})\|_F\big] = \frac{|\rho_{pq}|}{\sqrt{C}}\,\mathbb{E}\big[\|\mathbf{Q}_p^\top \mathbf{Q}_q\|_F\big] \leq |\rho_{pq}|\sqrt{\frac{C}{H}} \xrightarrow[H\to\infty]{} 0, \tag{24}$$

*so the rasterization-induced correlation $\rho_{pq}$ of Eq. 12 is suppressed for any two pixels whose canonical anchors differ, and increasingly so as the noise dimension $H$ grows relative to $C$.*

*Proof.* Immediate from Lemma 1 applied under Assumption 1 (with $\mathbf{U} = \mathbf{Q}_p, \mathbf{V} = \mathbf{Q}_q$), together with $\mathbf{Q}_p^\top \mathbf{Q}_p = \mathbf{I}_C$ from the reduced QR factorization (Eq. 15), and the definition $\operatorname{Cov}(\boldsymbol{\epsilon}_p^{\mathrm{3D}}, \boldsymbol{\epsilon}_q^{\mathrm{3D}}) = \rho_{pq}\mathbf{Q}_p^\top \mathbf{Q}_q$ of Eq. 18. □

**Remark 1.** *This argument explains* why *distinctness of the anchor helps, but not* how far apart *two anchors must be before Assumption 1 is a good approximation: two points much closer together than the shortest wavelength present in the fixed frequency vectors $\boldsymbol{\omega}_a$ still produce almost the same $\mathbf{A}_p \approx \mathbf{A}_q$ and $\mathbf{Q}_p \approx \mathbf{Q}_q$, so $\mathbf{Q}_p^\top \mathbf{Q}_q \approx \mathbf{I}_C$ instead of $\mathbf{0}$ – this is exactly why Sec. 3.4 also needs* high-frequency *harmonics, not merely* many *of them: frequency sets the length scale below which Assumption 1 stops holding and decorrelation degrades toward the pre-projection correlation $\rho_{pq}$. This is also why the occlusion-boundary case flagged in Sec. 3.4 is a genuine exception: the blended anchor of Eq. 13 can there be arbitrarily close to both surfaces' anchors, defeating the distinctness the argument relies on.*

**Scope.** This proposition addresses only the decorrelation induced by the canonical-position-dependent projection field. It does not establish exact whitening of the complete rasterized noise field, since shared primitives, occlusion-boundary blending, and visibility changes can preserve residual correlations. It also does not model correspondence errors in unobserved regions caused by the finite background proxy. Finally, Assumption 1 is an idealization: it treats the projection bases induced by distinct canonical positions as independent Haar-random subspaces, whereas in the actual method they are deterministic functions of the fixed frequencies $\boldsymbol{\omega}_a$, phases $\phi_{a,c}$, and scale $s$. The quality of this approximation therefore depends on the separation of canonical positions relative to the frequencies used by the projection field. Empirically, we found that H=128 works fine and we keep it constant through all the experiments.

### A.4 Noise Rasterization and View Consistency

Sec. 3.3 claims that two cameras observing the same 3D content receive the same rasterized noise because both rasterize the same canonical anchor. We prove this exactly, for *arbitrary* pairs of camera poses – not only the pure-rotation special case – and for the general 4D (dynamic) setting, where a deformation module has already produced the time-dependent Gaussian parameters. In 4D Gaussian Splatting this result applies independently at each timestamp, after the deformation module has produced the time-dependent parameters of the 3D Gaussian primitives; we therefore fix a single timestamp $t$ throughout and do not address temporal consistency here (that is a separate claim, discussed qualitatively in Sec. 3.4).

**Setup at a fixed time.** Consider two time-dependent Gaussian centers

$$\mathbf{p}(t), \mathbf{q}(t) \in \mathbb{R}^3. \tag{25}$$

These centers may be generated by a 4D deformation field $D$ from their canonical positions $\mathbf{p}_0, \mathbf{q}_0$ – i.e. $\mathbf{p}_0, \mathbf{q}_0$ play the role of the canonical anchors $\bar{\boldsymbol{\mu}}_i$ of Eq. 4:

$$\mathbf{p}(t) = D(\mathbf{p}_0, t), \qquad \mathbf{q}(t) = D(\mathbf{q}_0, t). \tag{26}$$

Let the camera-to-world pose at timestamp $t$ be

$$T_{wc}(t) = \begin{bmatrix} R_{wc}(t) & \mathbf{c}(t) \\ \mathbf{0}^\top & 1 \end{bmatrix} \in SE(3), \tag{27}$$

where $R_{wc}(t) \in SO(3)$ describes the camera orientation and $\mathbf{c}(t) \in \mathbb{R}^3$ denotes the camera center in world coordinates. The corresponding world-to-camera transformation is

$$T_{cw}(t) = T_{wc}(t)^{-1} = \begin{bmatrix} R_{cw}(t) & \mathbf{t}_{cw}(t) \\ \mathbf{0}^\top & 1 \end{bmatrix}, \qquad R_{cw}(t) = R_{wc}(t)^\top, \quad \mathbf{t}_{cw}(t) = -R_{cw}(t)\mathbf{c}(t). \tag{28}$$

The camera-space coordinate of $\mathbf{p}(t)$ is therefore

$$\mathbf{x}_p^c(t) = R_{cw}(t)\mathbf{p}(t) + \mathbf{t}_{cw}(t) = R_{cw}(t)\big(\mathbf{p}(t) - \mathbf{c}(t)\big), \tag{29}$$

and, similarly, $\mathbf{x}_q^c(t) = R_{cw}(t)\big(\mathbf{q}(t) - \mathbf{c}(t)\big)$. Let $\mathbf{K}$ denote the camera intrinsic matrix and $\pi_K$ the perspective projection; for a camera-space point $\mathbf{x} = [x, y, z]^\top$,

$$\pi_K(\mathbf{x}) = \begin{bmatrix} f_x \frac{x}{z} + c_x \\ f_y \frac{y}{z} + c_y \end{bmatrix}. \tag{30}$$

The projected pixel locations of $\mathbf{p}(t)$ and $\mathbf{q}(t)$ are thus

$$\mathbf{u}_p(t) = \pi_K\big(R_{cw}(t)(\mathbf{p}(t) - \mathbf{c}(t))\big), \qquad \mathbf{u}_q(t) = \pi_K\big(R_{cw}(t)(\mathbf{q}(t) - \mathbf{c}(t))\big). \tag{31}$$

Both are governed by the same world-to-camera transformation $T_{cw}(t)$, although their pixel locations generally differ, since perspective projection is depth- and position-dependent.

**Two cameras observing the same primitive.** Let $\Pi_1 = T_{wc}^{(1)}(t)$ and $\Pi_2 = T_{wc}^{(2)}(t)$ be *any* two camera poses at the same timestamp $t$ – not necessarily related by a rotation, or by any other closed-form relationship. By Eq. 31, the same primitive's center $\mathbf{p}(t)$ projects to generally different pixels $\mathbf{u}_p^{(1)}(t)$ under $\Pi_1$ and $\mathbf{u}_p^{(2)}(t)$ under $\Pi_2$. What is invariant is not the pixel location but what gets attached to it: the primitive's canonical anchor $\mathbf{p}_0$ and its identity noise vector $\mathbf{z}_i$ of Eq. 7 are attributes of the primitive itself, fixed once at construction time and never a function of $t$, of $\Pi_1$, or of $\Pi_2$ (Sec. 3.3).

**Assumption 2** (Well-resolved primitive)**.** *At pixel $p$ under $\Pi_1$, primitive $i$ dominates the rasterization weight: $w_{p,i}(\Pi_1) > 0$ and $w_{p,j}(\Pi_1) \approx 0$ for all $j \neq i$ (and symmetrically for pixel $p'$ under $\Pi_2$). This is the idealization underlying the statement in Sec. 3.4 that correspondence is exact when the same primitives dominate the rasterization; it fails at genuine multi-primitive blending, most notably at occlusion boundaries.*

**Proposition 2** (View consistency of the rasterized noise)**.** *Let $p$ be the pixel under $\Pi_1$ and $p'$ the pixel under $\Pi_2$ at which primitive $i$ is rasterized (i.e. $p = \mathbf{u}_p^{(1)}(t)$, $p' = \mathbf{u}_p^{(2)}(t)$ from Eq. 31). Under Assumption 2,*

$$\mathbf{X}_p(\Pi_1) = \mathbf{X}_{p'}(\Pi_2) = \mathbf{p}_0, \qquad \mathbf{Q}_p(\Pi_1) = \mathbf{Q}_{p'}(\Pi_2), \qquad \boldsymbol{\epsilon}_p^{\mathrm{3D}}(\Pi_1) = \boldsymbol{\epsilon}_{p'}^{\mathrm{3D}}(\Pi_2), \tag{32}$$

*for* any *two camera poses $\Pi_1, \Pi_2$ at the same timestamp $t$ – no rotation, translation, or other structural relationship between $\Pi_1$ and $\Pi_2$ is required.*

*Proof.* Under Assumption 2, the canonical-position rasterization of Eq. 13 collapses to the single dominant term: $\mathbf{X}_p(\Pi_1) = \frac{w_{p,i}(\Pi_1)\mathbf{p}_0}{w_{p,i}(\Pi_1)+\varepsilon} \to \mathbf{p}_0$ as $w_{p,i}(\Pi_1) \gg \varepsilon$, and identically $\mathbf{X}_{p'}(\Pi_2) \to \mathbf{p}_0$, giving the first equality in Eq. 32; this uses only that $\mathbf{p}_0$ does not depend on $\Pi_1$ or $\Pi_2$. Since $\mathbf{Q}_p$ is a fixed deterministic function of the canonical position alone (Eq. 15), equal anchors give $\mathbf{Q}_p(\Pi_1) = \mathbf{Q}(\mathbf{p}_0) = \mathbf{Q}_{p'}(\Pi_2)$. Similarly, the normalized feature of Eq. 11 collapses to $\widetilde{\mathbf{Y}}_p(\Pi_1) = \frac{w_{p,i}(\Pi_1)\mathbf{z}_i}{\sqrt{w_{p,i}(\Pi_1)^2+\varepsilon}} \to \mathbf{z}_i$ (using $w_{p,i}(\Pi_1) > 0$), independent of the specific value of $w_{p,i}(\Pi_1)$, and identically $\widetilde{\mathbf{Y}}_{p'}(\Pi_2) \to \mathbf{z}_i$; this uses only that $\mathbf{z}_i$ does not depend on $\Pi_1$ or $\Pi_2$ either. Combining the equal $\mathbf{Q}$ and $\widetilde{\mathbf{Y}}$ through Eq. 16 gives $\boldsymbol{\epsilon}_p^{\mathrm{3D}}(\Pi_1) = \mathbf{Q}(\mathbf{p}_0)^\top \mathbf{z}_i = \boldsymbol{\epsilon}_{p'}^{\mathrm{3D}}(\Pi_2)$, the last equality of Eq. 32. At no point did the proof use a relationship between $\Pi_1$ and $\Pi_2$, a closed-form pixel correspondence, or the specific values of $\mathbf{u}_p^{(1)}(t), \mathbf{u}_p^{(2)}(t)$ from Eq. 31 beyond their existence. □

**Corollary 1** (Pure rotation gives the correspondence in closed form)**.** *If $\Pi_2$ is obtained from $\Pi_1$ by a pure rotation $\mathbf{R} \in SO(3)$ about its own optical center (same $\mathbf{c}(t)$, $R_{cw}^{(2)}(t) = \mathbf{R}R_{cw}^{(1)}(t)$, i.e. $R_{wc}^{(2)}(t) = R_{wc}^{(1)}(t)\mathbf{R}^\top$, with $\mathbf{R}$ expressed in the first camera's frame), the pixel $p'$ at which Prop. 2 applies is additionally known in closed form, independent of $\mathbf{p}_0$'s depth: writing homogeneous pixel coordinates $\widetilde{\mathbf{u}}$,*

$$\widetilde{\mathbf{u}}' \propto \left(\mathbf{K}\mathbf{R}\mathbf{K}^{-1}\right)\widetilde{\mathbf{u}}. \tag{33}$$

*Proof.* Writing $\mathbf{x}^c = R_{cw}^{(1)}(t)(\mathbf{p}(t) - \mathbf{c}(t))$ and $z = [\mathbf{x}^c]_3 > 0$, so $\mathbf{x}^c = z\mathbf{K}^{-1}\widetilde{\mathbf{u}}$, the rotated camera's coordinate is $\mathbf{x}^{c'} = R_{cw}^{(2)}(t)(\mathbf{p}(t) - \mathbf{c}(t)) = \mathbf{R}\mathbf{x}^c = z\mathbf{R}\mathbf{K}^{-1}\widetilde{\mathbf{u}}$, so $\widetilde{\mathbf{u}}' \propto \mathbf{K}\mathbf{x}^{c'} = z\,\mathbf{K}\mathbf{R}\mathbf{K}^{-1}\widetilde{\mathbf{u}} \propto \mathbf{K}\mathbf{R}\mathbf{K}^{-1}\widetilde{\mathbf{u}}$, the depth $z$ being absorbed by the projective equivalence. Unlike Prop. 2, which needs no relationship between $\Pi_1, \Pi_2$ because it never has to locate $p'$ explicitly, this closed form exists only because a pure rotation about the optical center introduces no parallax (Hartley & Zisserman Hartley et al. (2003)); a general pair $\Pi_1, \Pi_2$ (translation included) has no such depth-independent map, which is exactly why Prop. 2 is stated without one. □

**Remark 2.** *Two points on scope.* (i) Where Assumption 2 fails: *at genuine multi-primitive blending – most notably a pixel straddling an occlusion boundary – $\mathbf{X}_p$ averages the anchors of unrelated surfaces (Eq. 13), so the clean collapse to $\mathbf{p}_0$ in the proof no longer holds exactly; this is the same exception already noted in Sec. 3.4 and in Prop. 1's remark.* (ii) Locus of the guarantee: *as with Props. 1, this is a statement about the* noise construction*, exact under Assumption 2; it does not by itself imply the trained diffusion model's generated frames inherit this consistency, which is what the view-synchronization metrics of Sec. 4.2 test empirically.*

Table 8: **Compute time breakdown** for the four main components of SyncVid. Times are reported in seconds per input clip with 75 frames at 480×832 resolution on an Nvidia H100 GPU. The DiT time is reported per denoising step.

| | 4D representation (NoPo4D) | Noise rasterization | Whitening | Video generation (WAN2.2-14B) | | |
|---|---|---|---|---|---|---|
| | | | | DiT | VAE decoder | Total |
| Compute time (sec.) | 4.18 | 6.44 | 0.53 | 23.97 / step | 4.60 | 1203.1 |

### A.5 Experimental Details

**Training.** We evaluate all baseline methods using their officially released checkpoints. We train SyncVid with the Wan2.2-14B backbone on eight H100 GPUs, using a per-GPU batch size of one and gradient accumulation over four steps, for an effective batch size of 32. The model is trained for 10K iterations with a learning rate of $1 \times 10^{-4}$.

**4D representation.** We use an off-the-shelf feed-forward Gaussian Splatting model, NoPo4D Balice et al. (2026), which avoids per-video reconstruction optimization and produces dynamic primitives that are active over finite temporal windows. Each primitive has a time-varying center, covariance, opacity, and appearance, together with a time-invariant canonical anchor. To render noise for views not covered by the source video, we model unobserved background regions with an enclosing cube-map: a finite, axis-aligned box centered at the centroid of the canonical anchors, with a half-extent set to a fixed multiple of the reconstructed scene extent. The cubemap therefore lies at a depth comparable to the scene rather than at infinity, as in an environment map. We index background texels by the ray–box intersection point instead of ray direction alone, allowing pixels that observe the same proxy surface to share a canonical location. The cubemap has six square faces, each containing $S \times S$ texels with $d$ channels; we use $S = 1024$ and $d = 3$.

**Inference.** Unless stated otherwise, we use 50 denoising steps for all SyncVid experiments, matching the default inference configuration of the WAN video models. We use the same fixed template as the text prompt during training and inference. Table 8 reports the runtime of the main SyncVid components. Constructing the 4D representation takes 4.18 seconds and is inexpensive relative to total generation time; because it is performed once per source video, this cost can also be amortized across multiple reshoots. The reported noise-rasterization time was measured on a particularly large dynamic scene containing approximately 6.53M Gaussian points. Video generation uses the original WAN configuration. More efficient step-distillation methods could reduce this cost, but evaluating them is beyond the scope of this work.

**Metrics.** For pixel matching (Pix. Mat.), we use RoMA-v2 Edstedt et al. to estimate point correspondences and count matches with uncertainty below $0.5$. We report the average number of matched pixels separately for dynamic and static regions. To identify dynamic foreground objects, we obtain a segmentation mask from the source video using SAM3 Carion et al. (2026). We use this mask to partition correspondences for each source–target frame pair, enabling separate evaluation of dynamic and static regions.

### A.6 Qualitative Comparison

For a more detailed qualitative comparison, we encourage readers to consult the HTML viewer provided in the supplemental materials. In the print version, Fig. 5 shows additional results from test scenes under a Dolly-out camera trajectory, while Figs. 6 and 7 show results under Arc-left and Arc-right trajectories, respectively. Each example contains six equally spaced frames from the source clip and from each method's generated reshoot.

### A.7 Limitations and Failure Examples

Although SyncVid is trained with varied errors in the 4D noise representation and is therefore less sensitive to reconstruction quality—the 4D representation is used primarily to rasterize a synchronized noise prior—it remains limited by the quality of the monocular reconstruction. Errors in geometry, tracking, or primitive re-instantiation can still appear as subject deformation or temporal drift, while disoccluded regions are only weakly constrained and may be hallucinated by the

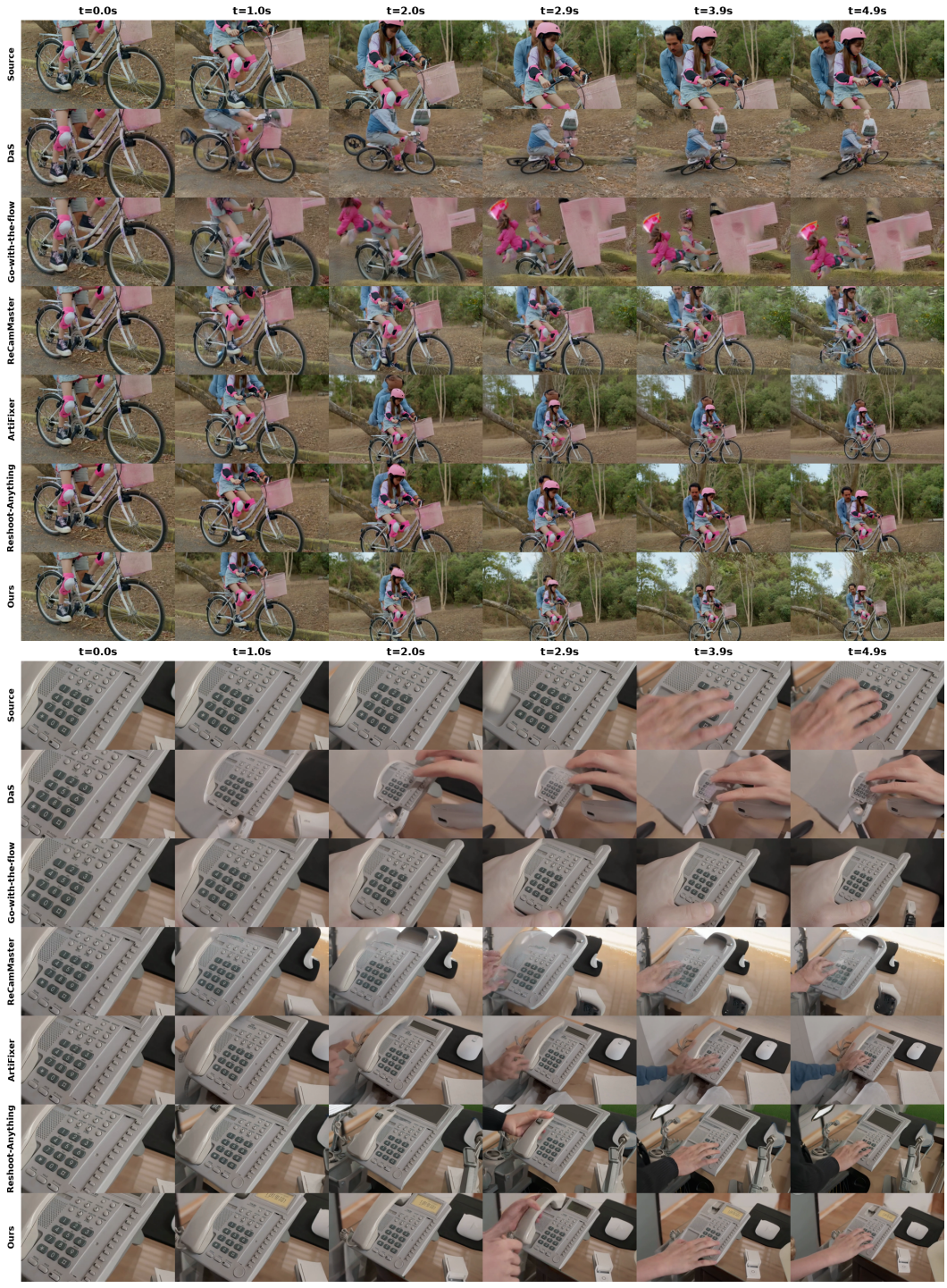


Figure 5: **Qualitative comparisons** under a Dolly-out camera trajectory. For each example, six equally spaced frames are shown for the source video and each method's generated reshoot.

diffusion model. Figure 8 illustrates these failure modes: the Arc-right example exhibits severe foreground inconsistencies, whereas the Dolly-out example retains the main subject and setting but introduces unrealistic content in newly exposed regions.

Although SyncVid excels on video-synchronization metrics and in preserving dynamic regions in the generated videos, it does not outperform state-of-the-art methods in camera-pose estimation;




Figure 6: **Qualitative comparisons** under an Arc-right camera trajectory. For each example, six equally spaced frames are shown for the source video and each method's generated reshoot.

methods such as ArtiFixer perform well on this task. We speculate that this gap is partly due to the absence of explicit camera-pose conditioning, such as the Plücker embeddings used by ArtiFixer. Providing this additional conditioning could improve pose accuracy, but integrating it is orthogonal to the structured-noise formulation presented here and is left for future work.

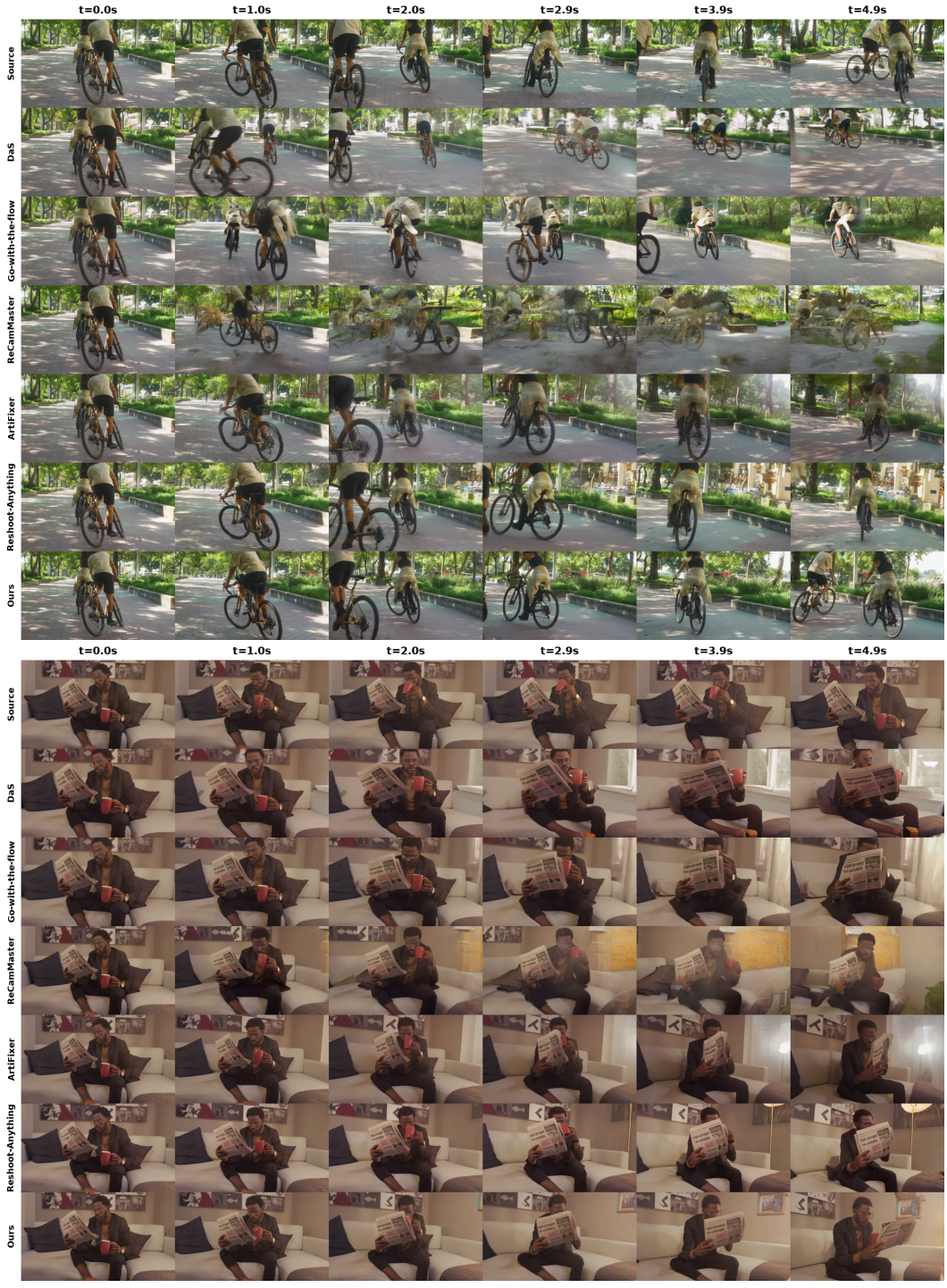


Figure 7: **Qualitative comparisons** under an Arc-left camera trajectory. For each example, six equally spaced frames are shown for the source video and each method's generated reshoot.

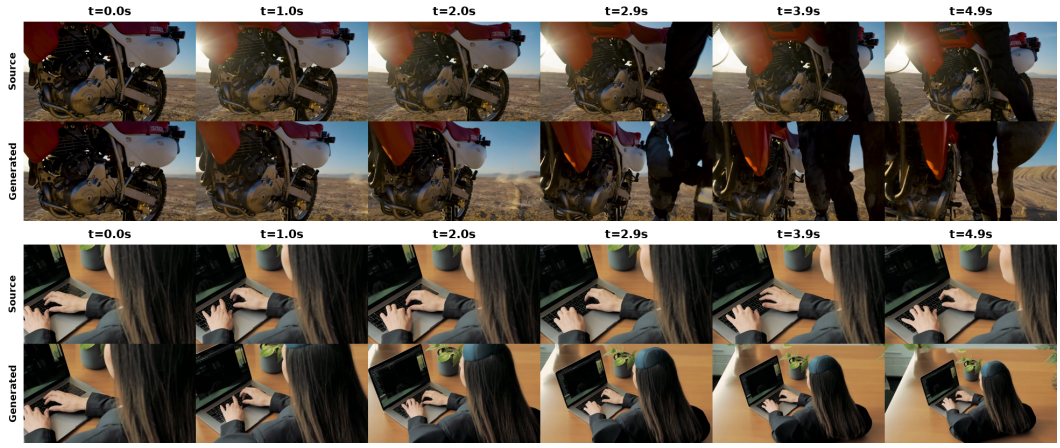


Figure 8: **Failure examples.** Each example shows six equally spaced source frames in the first row, followed by the corresponding generated frames. The top example follows Arc-left and the bottom example shows Dolly-out trajectory.