MoVE-4D: Feed-forward Video-to-4D Generation via 4D Structured Gaussians

Anonymous authors
Paper under double-blind review

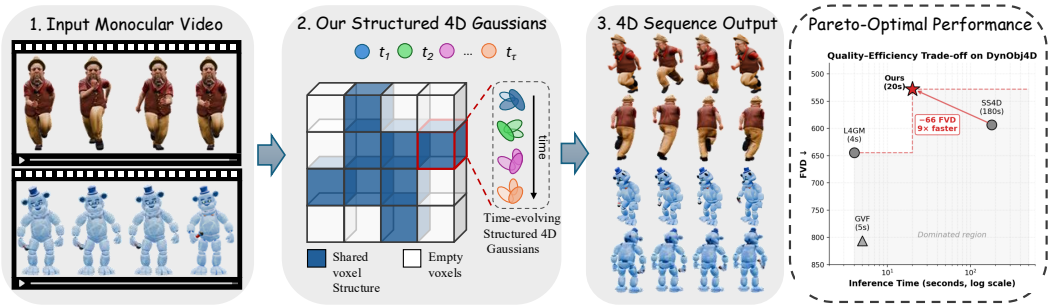


Figure 1: **MoVE-4D** for video-to-4D generation. From a monocular video, MoVE-4D generates a structured 4D representation where 4D Gaussians anchored on shared voxels evolve over time, producing high-fidelity 4D sequences at Pareto-optimal quality–efficiency.

Abstract

Generating dynamic 4D assets from monocular videos in a feed-forward manner has rapidly progressed by extending pretrained 3D foundation models to time. However, existing paradigms still treat 3D representations as the atomic unit and introduce temporal modeling externally, either through canonical-shape deformation or per-frame generation coupled by temporal attention. Both designs leave 3D structure unshared at the representation level, leading to limited motion expressivity in deformation-based methods and a persistent quality–efficiency trade-off in per-frame methods. We propose MoVE-4D, a feed-forward video-to-4D framework built on a 4D-native representation that shares 3D structure across time at the representation level itself. Specifically, we introduce structured 4D Gaussians on motion-occupied voxels, where sparse voxel anchors are reused across frames within a temporal window and time-dependent Gaussian attributes capture local dynamics. MoVE-4D generates this representation in two stages: it first predicts motion-occupied regions through voxel-wise occupancy merging, then populates them with structured 4D Gaussians decoded from a Compact Temporal Structured LATent (CT-SLAT) that aggregates multi-frame visual features anchored at the same voxel. Extensive experiments show that MoVE-4D delivers a Pareto-optimal quality–efficiency trade-off, surpassing existing feed-forward methods in visual quality and spatial-temporal consistency.

1 Introduction

Video-to-4D single-object generation aims to reconstruct a dynamic 3D asset from a monocular video, benefiting applications such as dynamic content creation, free-viewpoint rendering, and AR/VR. Previous methods (Bahmani et al., 2024; Ren et al., 2023; Jiang et al., 2023; Chu et al., 2024; Ling et al., 2024; Zhang et al., 2024) rely on score distillation sampling (Poole et al., 2022) (SDS) from image/video/3D generative models and per-scene optimization, which suffer from heavy computation and unstable quality. Recent feed-forward methods (Wu et al., 2025; Sabathier et al., 2026; Yenphraphai et al., 2025; Zhang et al., 2025a; Li et al., 2025b; Ren et al., 2024) predict 4D

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.

assets in a single end-to-end pass, substantially improving efficiency and pushing 4D generation toward practical use.

Specifically, these feed-forward methods build on pretrained 3D foundation models (Xiang et al., 2025; Li et al., 2025a; Tang et al., 2024) for geometric capacity, and differ mainly in how they expand 3D priors to the temporal dimension. One line of methods (Wu et al., 2025; Zhang et al., 2025a; Chen et al., 2026) follows a **3D-anchored** paradigm, where a base 3D asset is first constructed from generated multi-view images or an explicit 3D representation, and the 4D sequence is then reconstructed by animating or deforming this base asset. These methods are often efficient, since the main spatial structure has already been established. However, they are also sensitive to errors in the initial 3D asset, which can be propagated and amplified during temporal reconstruction, leading to accumulated artifacts and implausible deformations.

Another line of methods follows a **3D-sequenced** (Li et al., 2025b; Ren et al., 2024) generation paradigm. Instead of relying on a separately reconstructed base 3D asset, these methods leverage the spatial priors of 3D foundation models and introduce temporal modules to generate time-varying 3D representations across the sequence. This avoids error accumulation from an explicit base asset and enables rich per-timestep 3D modeling. However, temporal relations are usually modeled implicitly, *e.g.*, through temporal attention, while similar geometry and appearance are still represented repeatedly at different timesteps. As a result, the model spends computation and capacity regenerating shared content rather than reusing it across time, limiting generation efficiency. Although stronger temporal downsampling can encourage temporal sharing, it may discard fine-grained temporal variations and degrade dynamic details. This suggests that efficient yet expressive 4D generation requires more than structural downsampling alone. It also requires a 4D-native representation whose compressed latent space remains compact while preserving the ability to share temporal information across frames.

We instantiate this principle through **structured 4D Gaussians** on **motion-occupied voxels**. Specifically, the 3D space is partitioned into voxel cells, and cells visited by the object within a temporal window are identified as **motion-occupied regions**, forming sparse spatial anchors shared across frames. **4D Gaussian primitives** (Duan et al., 2024; Yang et al., 2024), a native 4D representation are placed on these anchors and extend standard 3D Gaussians (Kerbl et al., 2023) into the temporal dimension, allowing the same primitive to span multiple timesteps. This design combines spatial sharing with temporal flexibility: a single voxel can host multiple primitives that evolve (activate, deform, fade) through time-dependent attributes, while the underlying voxel structure remains composed of shared anchors reused across frames.

Generating this representation from a monocular video raises two coupled problems: identifying the motion-occupied regions, and populating them with 4D Gaussians that recover the dynamic content. We address them with a two-stage pipeline, **MoVE-4D** (**Mo**tion-occupied **V**oxels with **E**volving **4D** Gaussians), reflecting our core design where motion-occupied voxels provide shared spatial supports and 4D Gaussian primitives evolve within them through time-dependent attributes. The first stage generates **motion-occupied regions** as compact spatial supports shared across time. The second stage generates structured 4D Gaussians on these supports through a Compact Temporal Structured LATent (CT-SLAT), which aggregates multi-frame visual features anchored at the same voxel into a single latent. Together, these designs anchor shared 3D structure on a common voxel grid while letting time-dependent attributes recover per-frame variation. Each stage is realized as a variational autoencoder (VAE) that learns a compact latent space for its target representation, paired with a video-conditioned rectified flow generator that predicts the latent. To avoid excessive temporal aggregation, we apply both stages over short temporal windows; temporal attention in the generators exchanges information across windows for coherent generation. Extensive experiments on held-out test sets demonstrate that MoVE-4D achieves a Pareto-optimal balance of quality and efficiency, delivering superior visual quality and spatial-temporal consistency while being substantially faster than methods of comparable quality.

- We propose structured 4D Gaussians on motion-occupied voxels as a **native 4D representation** that shares 3D structure across time at the representation level, and realize it through CT-SLAT, a compact temporal structured latent that aggregates multi-frame visual evidence into time-evolving 4D Gaussian primitives.
- We design MoVE-4D, a **two-stage feed-forward framework** that first generates motion-occupied regions and then populates them with structured 4D Gaussians, enabling efficient video-to-4D generation that breaks the quality–efficiency trade-off of existing paradigms.

- We curate a **4D benchmark** and show that MoVE-4D achieves Pareto-optimal performance, outperforming feed-forward baselines in quality while remaining substantially faster than methods of comparable quality.

2 Related Work

Feed-forward 3D Generation. Feed-forward 3D generation methods (Tang et al., 2024; Xiang et al., 2025; Li et al., 2025a; Zhang et al., 2023; Chen et al., 2025) learn direct mappings from multi-modal inputs to explicit 3D assets. Early methods like LGM (Tang et al., 2024) introduce a multi-view Gaussian feature backbone and an asymmetric U-Net for fast 3D Gaussian generation from multi-view images. Recent diffusion-based (Song et al., 2020; Esser et al., 2024) methods further improve generation quality along two main directions. One direction focuses on high-quality explicit 3D representations, such as TRELLIS (Xiang et al., 2025), which separately generates 3D structure and appearance for 3D Gaussian generation. Another direction emphasizes geometry generation, often producing mesh-based assets (Li et al., 2025a; Zhang et al., 2023; Chen et al., 2025; Zhao et al., 2025). These methods provide strong 3D priors and generation backbones for feed-forward 4D generation.

Implicit 4D Representation. Implicit 4D representations generally fall into two categories: deformable 3D representations (Wang et al., 2023; Pumarola et al., 2021; Das et al., 2024; Lu et al., 2024; Wu et al., 2024) and native 4D representations (Yang et al., 2024; Duan et al., 2024; Katsumata et al., 2024; Zhang et al., 2025b). The former models dynamics by deforming a canonical 3D representation over time. The latter, represented by 4DGS, directly parameterizes dynamic content in the joint space-time domain. While deformable 3D representations impose stronger structural constraints, native 4D representations provide greater flexibility and better support representation-level sharing across time. In this work, we adopt native 4DGS for its flexible space-time modeling ability, which helps reduce error accumulation in 3D-anchored methods and improve efficiency over 3D-sequenced methods by reusing shared information across time.

Feed-forward 4D Generation. Feed-forward 4D generation typically relies on strong 3D priors to reduce the difficulty of dynamic generation. These priors can be introduced explicitly, for example by starting from given 3D assets, 3D observations, or regressed 3D sequences, which leads to a 3D-anchored generation paradigm (Wu et al., 2025; Zhang et al., 2025a). Alternatively, they can be introduced implicitly by leveraging pretrained 3D foundation models and lifting their spatial understanding to the temporal dimension, as in (Li et al., 2025b). From the perspective of generated content, existing methods also differ in their emphasis. Some methods (Wu et al., 2025; Sabathier et al., 2026; Yenphraphai et al., 2025) focus primarily on geometry-centric generation, often producing mesh-based dynamic assets while paying relatively less attention to appearance fidelity. Others (Zhang et al., 2025a; Li et al., 2025b; Ren et al., 2024) target appearance-centric 4D generation with implicit 4D representations, aiming for high-quality novel view synthesis of dynamic objects. This work focuses on the latter setting, with the goal of improving generation efficiency without sacrificing, or even improving, the generation quality through a compact 4D-native representation.

3 Method

Given a monocular object video $\mathcal{V}$, the goal is to generate a T-frame 4D sequence represented by structured 4D Gaussians $\mathcal{G}_{4D}$. We first extract per-frame visual features from $\mathcal{V}$, using a pretrained DINO-V2 (Oquab et al., 2023) encoder, which serve as conditioning signals injected into both stages of our pipeline through cross-attention. To achieve high-quality generation results, we adopt a two-stage generation pipeline, where each stage learns an autoencoder for the target representation and a video-conditioned generator for latent prediction. In the first stage, the generator predicts a latent code z_o, which is decoded into motion occupied region by merging occupancies over a specific time period. This captures a coarse region occupied by the object over time. In the second stage, conditioned on the motion occupancy, the generator predicts CT-SLAT, a compact latent constructed by aggregating multi-timestep visual features onto active motion-occupied voxels. The predicted CT-SLAT is then decoded into structured 4D Gaussians $\mathcal{G}_{4D}$.

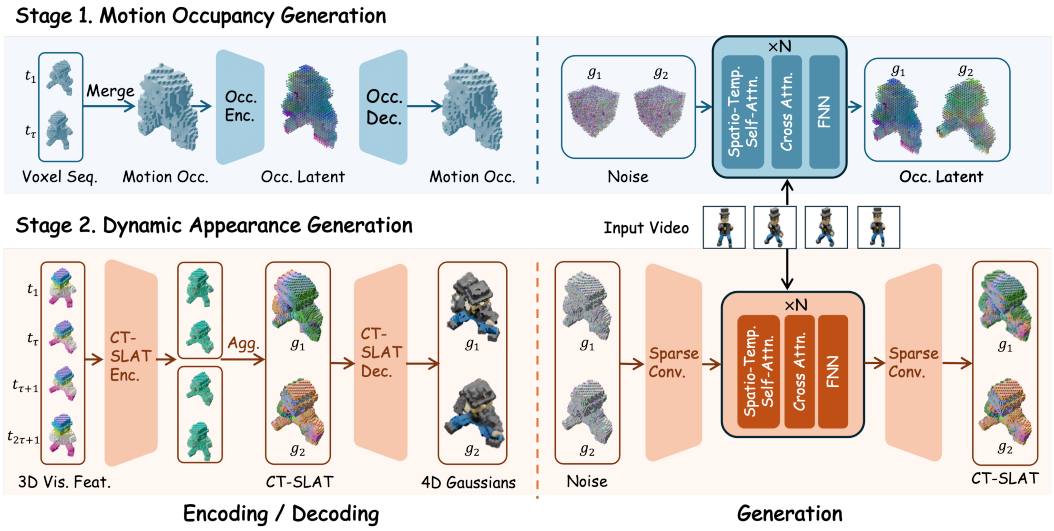


Figure 2: Overview of MoVE-4D. The input video is partitioned into temporal groups (e.g., g_1, g_2), each spanning τ consecutive frames $\{t_i, \ldots, t_{i+\tau-1}\}$. Each stage consists of a sparse VAE that encodes each group into a per-group latent (left of dashed line), and a video-conditioned generator (right) that jointly predicts all latents via spatio-temporal self-attention across groups, with input video features injected through cross-attention. Stage 1 generates the shared 4D structure as motion-occupied voxels merged across time; Stage 2 generates the time-evolving content via CT-SLAT, which aggregates multi-frame visual features on these voxels and decodes into structured 4D Gaussians. By separating shared structure from local dynamics, MoVE-4D generates 4D content where 3D structure is reused across time at the representation level.

3.1 Structured 4D Gaussians

Structured 4D Gaussians organize 4D Gaussian primitives (Yang et al., 2024; Duan et al., 2024) over motion-occupied regions. We partition the 3D space into a voxel grid, where motion occupancy $\mathcal{M}$ marks the cells visited by the object over time. 4D Gaussian primitives $\mathcal{G}_{4D}$ are then attached to these occupied cells, forming a sparse, cell-based representation that is easier to encode and generate than a freely optimized, unstructured Gaussian set. This also allows dynamic content to be modeled within shared spatial supports for redundancy reduction.

4D Gaussians. A 4D Gaussian primitive extends a standard 3D Gaussian with an additional temporal dimension. Each primitive is parameterized as $\mathcal{G}_{4D} = \{\boldsymbol{\mu}_{4D}, \boldsymbol{\Sigma}_{4D}, o_{4D}, \mathbf{h}_{4D}\}$, where $\boldsymbol{\mu}_{4D} = [\boldsymbol{\mu}_{3D}^\top, \mu_t]^\top$ denotes the space-time center, with $\boldsymbol{\mu}_{3D}$ and μ_t being the spatial and temporal centers, respectively. $\boldsymbol{\Sigma}_{4D}$ denotes the space-time covariance, while o_{4D} and $\mathbf{h}_{4D}$ denote the opacity and color coefficients. At timestamp t, the primitive is sliced along the temporal dimension into a 3D Gaussian, following (Yang et al., 2024), $\mathcal{G}_{3D}^{(t)} = \{\boldsymbol{\mu}_{3D}^{(t)}, \boldsymbol{\Sigma}_{3D}^{(t)}, o^{(t)}, \mathbf{h}^{(t)}\}$, which is rendered using standard 3D Gaussian splatting. This formulation allows each primitive to span multiple timesteps and represent 4D data in a spatially and temporally compact form.

Temporal Grouping. Ideally, a full 4D sequence of length T can be represented by a single pair $(\mathcal{M}, \mathcal{G}_{4D})$. In practice, we adopt temporal grouping: the sequence is uniformly partitioned into T_G temporal groups, each corresponding to a subsequence of length τ. For a temporal group g, which covers timesteps $\mathcal{T}_g = \{(g-1)\tau, \ldots, g\tau - 1\}$, we represent it with its own motion occupancy and 4D Gaussian representation, denoted as $(\mathcal{M}^{(g)}, \mathcal{G}_{4D}^{(g)})$. The full sequence is therefore represented by $\{(\mathcal{M}^{(g)}, \mathcal{G}_{4D}^{(g)})\}_{g=1}^{T_G}$[1]. During rendering, the 4D Gaussians from all groups are merged to form the complete sequence representation. This design keeps temporal aggregation local while preserving representation fidelity.

[1]For notation simplicity, we omit the spatial index of Gaussians attached to individual motion-occupied voxels.

3.2 MOTION OCCUPANCY GENERATION

Formally, $\mathcal{M}$ is constructed by merging timestep-wise voxel occupancies over a temporal group. Given a 4D sequence, we first compute voxel occupancy at each timestep as $\mathcal{O}_t \in \{0,1\}^{N^3}$, $N = 64$. For a temporal group, the group-level motion occupancy is defined as $\mathcal{M}^{(g)} = max_{t \in \mathcal{T}_g}(\mathcal{O}_t)$ where $\max(.)$ denotes the element-wise maximum operation for motion occupancy merging, recording the spatial extent swept by the object within the group, providing compact structural guidance for generation while preserving a coarse geometry about the 4D motion.

Motion occupancy VAE. We adopt an occupancy VAE (Xiang et al., 2025; Li et al., 2025b) to learn a compact latent space representing group-level motion occupancy. The VAE is applied independently to each temporal group. Given $\mathcal{M}^{(g)}$, the encoder $\mathcal{E}_o$ encodes it into the latent code $z_o^{(g)}$. The decoder $\mathcal{D}_o$ reconstructs the corresponding motion occupancy as $\widehat{\mathcal{M}}^{(g)} = \mathcal{D}_o(z_o^{(g)})$. The VAE is trained with a Dice reconstruction loss and KL regularization:

$$\mathcal{L}_{\text{occ}} = \sum_{g=1}^{T_G} \left[\mathcal{L}_{\text{Dice}}(\widehat{\mathcal{M}}^{(g)}, \mathcal{M}^{(g)}) + \lambda_o D_{\text{KL}} \left(q_o(z_o|\mathcal{M}^{(g)}) \,\|\, \mathcal{N}(\mathbf{0}, \mathbf{I}) \right) \right], \tag{1}$$

where $\mathcal{L}_{\text{Dice}}(\widehat{\mathcal{M}}, \mathcal{M}) = 1 - \frac{2\sum_{\mathbf{v}} \widehat{\mathcal{M}}_{\mathbf{v}} \mathcal{M}_{\mathbf{v}}}{\sum_{\mathbf{v}} \widehat{\mathcal{M}}_{\mathbf{v}} + \sum_{\mathbf{v}} \mathcal{M}_{\mathbf{v}}}$, with $\mathbf{v}$ indexing voxels. The learned latent codes $\{z_o^{(g)}\}_{g=1}^{T_G}$ provide compact representations of the motion-occupied structures for the generator.

Motion occupancy generator. After training the motion occupancy VAE, we encode each training sequence group by group, obtaining a latent sequence $\mathbf{Z}_o = \{\mathbf{z}_o^{(g)}\}_{g=1}^{T_G}$. We then train a video-conditioned rectified flow transformer (Esser et al., 2024) to generate $\mathbf{Z}_o$ from the input monocular video $\mathcal{V}$. To model dependencies across temporal groups, we augment the transformer with expanded temporal attention and temporal RoPE. Let $\mathbf{H}_o \in \mathbb{R}^{(B\ T_G) \times M \times C}$ denote the intermediate tokens before a temporal attention layer, where B is the batch size, M is the number of tokens per group, and C is the feature dimension. We apply the temporal attention layer (Yao et al., 2025; Li et al., 2025b) γ_t by reshaping the tokens as

$$\begin{aligned} \widetilde{\mathbf{H}}_o &= \gamma_t(\text{reshape}(\mathbf{H}_o, (B\ T_G)\ M\ C \to (B\ M)\ T_G\ C)), \\ \mathbf{H}'_o &= \text{reshape}(\widetilde{\mathbf{H}}_o, (B\ M)\ T_G\ C \to (B\ T_G)\ M\ C). \end{aligned} \tag{2}$$

where γ_t performs attention along the group dimension with temporal RoPE. This allows tokens with the same spatial index to exchange information across temporal groups while preserving the original group-wise layout. The temporal attention module allows the generator to exchange information across group-level latents. We train the generator under a rectified flow objective. Given a noise sequence $\boldsymbol{\epsilon}_o$ and the target latent sequence $\mathbf{Z}_o$, we define $\mathbf{Z}_{o,r} = (1-r)\boldsymbol{\epsilon}_o + r\mathbf{Z}_o$, where $r \in [0,1]$. The loss is calculated as:

$$\mathcal{L}_{\text{flow}}^{o} = \mathbb{E}_{r,\boldsymbol{\epsilon}_o,\mathbf{Z}_o} \left[\|g_o(\mathbf{Z}_{o,r}, r, \mathcal{V}) - (\mathbf{Z}_o - \boldsymbol{\epsilon}_o)\|_2^2 \right]. \tag{3}$$

At inference time, the learned flow is integrated from noise to obtain $\mathbf{Z}_o = \{z_o^{(g)}\}_{g=1}^{T_G}$, and each latent is decoded into the predicted motion occupancy $\mathcal{M}^{(g)} = \mathcal{D}_o(\hat{z}_o^{(g)})$.

3.3 DYNAMIC APPEARANCE GENERATION

Given the group-level motion occupancy $\mathcal{M}^{(g)}$, we construct CT-SLAT on the motion occupied voxels using a temporal asymmetric VAE, *i.e*, the encoder extracts timestep-wise visual features, while the decoder operates on grouped CT-SLAT features and reconstructs structured 4D Gaussians for each temporal group.

CT-SLAT VAE. Following (Xiang et al., 2025), for each timestep t, we extract DINOv2 (Oquab et al., 2023) features from multi-view images and project each active voxel onto these feature maps to obtain 3D visual features. We then process these per-frame sparse voxel features with an encoder equipped with temporal attention and Swin-window attention, producing encoded features $\mathbf{H}^{(t)}$. Since each timestep may contain a different set of occupied voxels, the encoded voxel features are not naturally aligned across time. A voxel in the group-level motion occupancy $\mathcal{M}^{(g)}$ may

appear at some timesteps and be absent at others. We therefore align the encoded features to the support of $\mathcal{M}^{(g)}$ and pad missing timestep features with zeros, yielding a fixed-length temporal feature sequence for each active voxel. Finally, an MLP-based aggregator $\mathcal{A}(.)$ compresses the τ encoded timestep features into a single CT-SLAT feature for each active voxel $\mathbf{Z}_s^{(g)} = \mathcal{A}\{\bar{\mathbf{H}}_g\}$. The resulting CT-SLAT compresses multi-view and multi-frame visual evidence into a structured latent representation attached to the active motion-occupied voxels. The decoder $\mathcal{D}_S$ maps the group-level CT-SLAT and motion occupancy to structured 4D Gaussian primitives $\mathcal{G}_{4D}^{(g)} = \mathcal{D}_S(\mathbf{Z}_s^{(g)}, \mathcal{M}^{(g)})$. To capture local geometry and appearance variations, each active voxel is assigned $K = 32$ 4D Gaussian primitives. After decoding all groups, we merge their primitives into a unified 4D representation $\mathcal{G}_{4D} = \bigcup_{g=1}^{T_G} \mathcal{G}_{4D}^{(g)}$ and supervise it through differentiable rendering over the full sequence:

$$\mathcal{L}_{\text{GS}} = \sum_{t=1}^{T} \sum_{c \in \mathcal{C}} \mathcal{L}_{\text{img}} \left(\mathcal{R}(\mathcal{G}_{4D}, c, t), I_{t,c} \right), \tag{4}$$

where $\mathcal{R}$ denotes Gaussian rendering, $c \in \mathcal{C}$ indicates the camera parameters, and $\mathcal{L}_{\text{img}}$ measures image reconstruction quality.

Video-conditioned CT-SLAT generator. After training the CT-SLAT VAE, we construct group-level CT-SLAT targets for each training sequence, denoted as $\mathbf{Z}_s = \{\mathbf{Z}_s^{(g)}\}_{g=1}^{T_G}$. We then train a second video-conditioned generator to predict $\mathbf{Z}_s$ from the input video $\mathcal{V}$ and the predicted $\{\mathcal{M}^{(g)}\}_{g=1}^{T_G}$. The generator is implemented as a sparse transformer based rectified flow model (Xiang et al., 2025; Esser et al., 2024), equipped with sparse temporal attention to exchange information across temporal groups while preserving the sparse voxel structure. Similar to the motion occupancy generator, given a noise sequence $\boldsymbol{\epsilon}_s$ and the target CT-SLAT sequence $\mathbf{Z}_s$, we define $\mathbf{Z}_{s,r} = (1-r)\boldsymbol{\epsilon}_s + r\mathbf{Z}_s$, where $r \in [0, 1]$. The generator is trained to predict the flow direction:

$$\mathcal{L}_{\text{flow}}^{s} = \mathbb{E}_{r, \boldsymbol{\epsilon}_s, \mathbf{Z}_s} \left[\| g_s(\mathbf{Z}_{s,r}, r, \mathcal{V}, \mathcal{M}) - (\mathbf{Z}_s - \boldsymbol{\epsilon}_s) \|_2^2 \right]. \tag{5}$$

At inference time, the learned flow is integrated from noise to obtain $\mathbf{Z}_s = \{\mathbf{Z}_s^{(g)}\}_{g=1}^{T_G}$. Each group-level CT-SLAT is decoded into structured 4D Gaussians as $\mathcal{G}_{4D}^{(g)} = \mathcal{D}_S(\mathbf{Z}_s^{(g)}, \mathcal{M}^{(g)})$.

4 EXPERIMENT

4.1 IMPLEMENTATION DETAILS

Training datasets. We train MoVE-4D on a curated dataset termed as DYNOBJ4D from ObjaverseXL (Deitke et al., 2023a) and Objaverse (Deitke et al., 2023b), containing about 20K animated object sequences. Each sequence contains 24 frames, which lies within the generation-length range supported by previous methods. We render the data using Blender EEVEE Next. For each animation sequence, we render 40 views. The first six views are fixed canonical views, including front, back, left, right, top, and bottom, while the remaining 34 views are randomly sampled following (Zhang et al., 2025a). We also follow their three lights setup consisting of one key point light and two area lights placed above and below the object. To ensure sufficient motion diversity, we select clips based on the maximum object motion range, which increases the effective motion amplitude and reduces static training samples.

Parameter configuration. The proposed pipeline contains four models: the motion occupancy VAE and generator, and the CT-SLAT VAE and generator. The models are trained using AdamW with a learning rate of 1×10^{-4}. We conduct our main experiments with $\tau = 8$. For 24-frame video generation, this yields $T_G = 3$ temporal groups[2]. The CT-SLAT VAE and generators are trained with batch size 1 and 2, respectively. Training is conducted on 8 H100 GPUs for one week.

Benchmarks. Our benchmark consists of three held-out test sets. 1) DYNOBJ4D contains 183 animated objects drawn from the publicly available Objaverse test split for evaluating novel-view synthesis quality, filtered to retain only those with substantial motion amplitude. For each object, we

[2]Temporal grouping does not introduce noticeable artifacts at group boundaries in the evaluated sequences, please see the appendix for more analysis.

use one front view as the monocular input and render ground-truth videos from four distinct fixed novel views, together with 180° and 360° orbit trajectories. The selection ensures no overlap with existing training or test sets, preventing contamination. 2) ACTIONBENCH (Sabathier et al., 2026) contains 128 dynamic objects with paired videos and point cloud sequences for evaluating generated geometry quality. 3) DAVIS4DTEST contains 20 dynamic objects selected from DAVIS (Perazzi et al., 2016) with only single-view RGB video sequences for evaluating real-world generalization ability.

Baselines and Metrics. Following previous works (Li et al., 2025b; Ren et al., 2024), we report CLIP, PSNR, SSIM, and LPIPS to evaluate image quality, and FVD (Unterthiner et al., 2018) to evaluate temporal consistency. We adopt CD-3D and CD-4D (Sabathier et al., 2026) to evaluate geometry quality. For Gaussian-based methods, we convert Gaussian centers into point clouds for geometry evaluation. For DAVIS4DTEST, where multi-view ground truth is unavailable, we conduct a user study to assess video quality and temporal dynamics. We compare our method with feed-forward 4D Gaussian generation frameworks, including L4GM (Ren et al., 2024), GVF (Zhang et al., 2025a), and SS4D (Li et al., 2025b), as well as mesh-based 4D generation methods for completeness, including Mesh4D (Jiang et al., 2026), Motion-3-to-4 (Chen et al., 2026), and ActionMesh (Sabathier et al., 2026)[3]. We use publicly available checkpoints for all baselines. We report the average of all views. All runtimes are measured on a single A6000 GPU for 24-frame inputs under the same setting.

4.2 QUANTITATIVE COMPARISON

Table 1: **Quantitative comparison on synthetic benchmarks.** Results on (a) DYNOBJ4D and (b) ACTIONBENCH, where cell colors mark the 1st, 2nd, and 3rd, per column.

(a) DYNOBJ4D

Method	CLIP↑	PSNR↑	SSIM↑	LPIPS↓	FVD↓	Time↓
L4GM	0.905	17.526	0.891	0.119	644.576	**4 s**
GVF	0.905	16.331	0.885	0.122	806.514	5 s
SS4D	0.926	16.920	0.890	0.105	593.568	180 s
Mesh4D	0.885	17.390	0.891	0.112	904.002	110 s
Motion 3-to-4	0.907	15.955	0.880	0.119	800.685	126 s
Ours	**0.935**	**18.525**	**0.900**	**0.090**	**527.888**	20 s

(b) ACTIONBENCH

Method	CD-3D↓	CD-4D↓
L4GM	0.072	0.109
GVF	0.083	0.122
SS4D	0.066	0.089
Mesh4D	0.062	0.097
Motion 3-to-4	0.065	0.105
ActionMesh	**0.054**	**0.086**
Ours	0.058	0.087

Synthetic Benchmark. As shown in Table 1, on DYNOBJ4D, our method achieves the best performance across all reported appearance metrics, outperforming both Gaussian-based baselines and recent mesh-based methods, including Mesh4D (Jiang et al., 2026) and Motion 3-to-4 (Chen et al., 2026). On ACTIONBENCH, our method achieves the second-best CD-3D and CD-4D scores, approaching the geometry-specialized ActionMesh (Sabathier et al., 2026), while additionally producing directly renderable, appearance-aware 4D Gaussian representations. These results demonstrate that MoVE-4D provides a favorable balance between geometry fidelity and appearance quality.

Table 2: **User study on** DAVIS4DTEST**.** Preference rates (%) for novel-view visual quality (RQ1), novel-view consistency (RQ2), and temporal coherence (RQ3).

Method	RQ1	RQ2	RQ3
L4GM	13.2	9.6	10.0
GVF	11.2	12.0	8.4
SS4D	18.8	18.0	18.8
Motion 3-to-4	19.2	17.2	20.0
Mesh4D	14.0	15.6	19.6
Ours	**23.6**	**27.6**	**23.2**

Real-world Benchmark. We further evaluate generalization to real-world videos on DAVIS4DTEST, as shown in Table 2. Since multi-view ground truth is unavailable for DAVIS videos, we conduct a user study comparing our method with the baselines. We randomly selected 10 of the 20 objects in DAVIS4DTEST, covering half of the benchmark for each participant. For each object, participants viewed the input video together with four novel fixed-view renderings and a 360° orbit video generated by all methods. The method identities were hidden, and the presentation order was randomized to avoid bias. After receiving brief instructions, participants answered three questions for each randomly presented example: 1) **RQ1**: Novel-view visual quality: Which result produces the highest-quality novel-view renderings, with the most realistic appearance and fewest visual artifacts? 2) **RQ2**: Novel-view consistency: Which result best preserves the object's identity, geometry, and appearance from

[3]ActionMesh does not provide a pipeline for generating textured dynamic meshes; therefore, we evaluate it only on ActionBench.

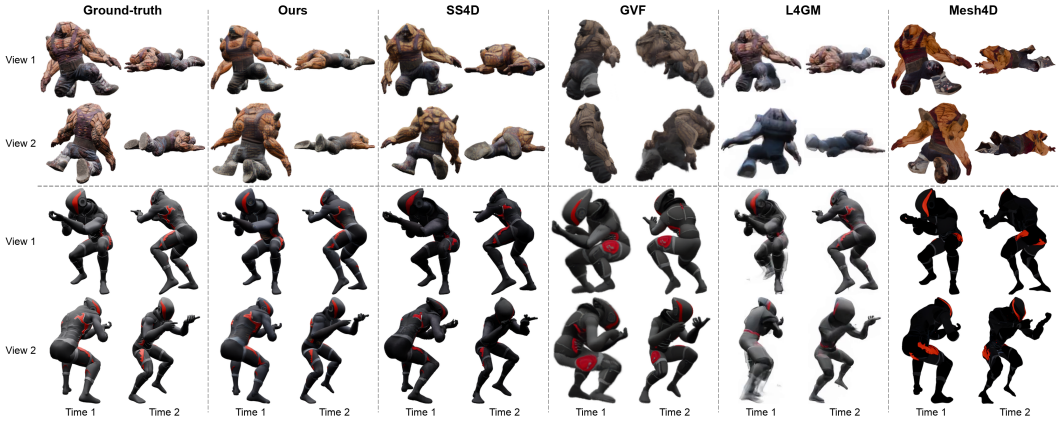


Figure 3: Qualitative comparison on synthetic dataset (DYNOBJ4D) compared with previous state-of-the-art methods.

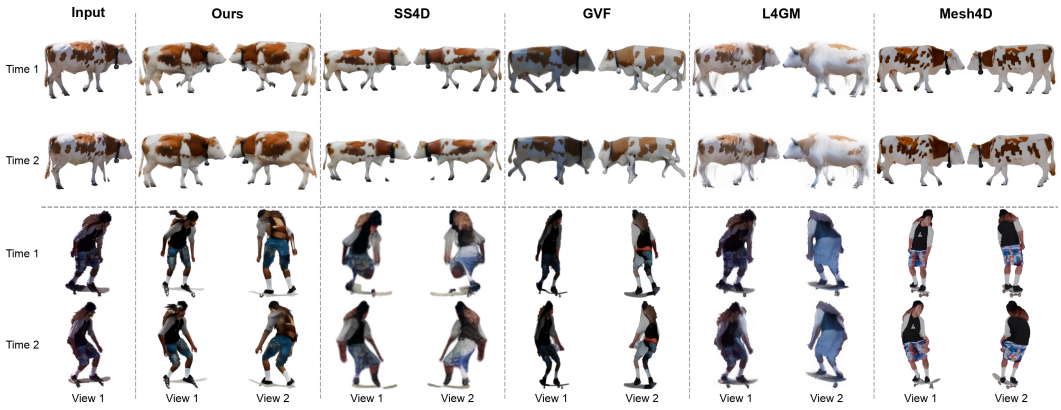


Figure 4: Qualitative comparison on real-world dataset (DAVIS4DTEST) compared with previous state-of-the-art methods.

the input video when viewed from novel viewpoints? 3) **RQ3**: Temporal coherence: Which result exhibits the most coherent dynamics over time, with the least flickering, blurring, abrupt deformation, or temporal discontinuity? We collected 750 valid responses from 25 participants with backgrounds in computer science and related fields, and Table 2 reports the selection rate for each method. These results provide complementary evidence that our method better preserves the input content while producing more coherent geometry and dynamics in out-of-distribution real-world cases.

4.3 QUALITATIVE COMPARISON

Synthetic Benchmark[4]**.** As shown in Fig. 3, our method generates more accurate object geometry and produces cleaner dynamic structures compared with prior feed-forward methods. In particular, the 3D-anchored representations of Mesh4D (Jiang et al., 2026) and GVF (Zhang et al., 2025a) lead to noticeable artifacts and unnatural deformations under complex dynamics, lacking architectural support for cross-frame reuse. The 3D-sequenced representation of L4GM (Ren et al., 2024) and SS4D (Li et al., 2025b) shows artifacts in appearance. Furthermore, even under compressed motion occupancy, our generated 4D assets maintain faithful shapes and coherent details, suggesting that the proposed representation effectively retains essential geometric and dynamic appearance information.

Real-world Benchmark[4]**.** Fig. 4 further shows results on real-world DAVIS videos. Although these inputs differ from the synthetic training distribution, our method still reconstructs plausible dynamic 4D assets with consistent object structure and reasonable motion. This demonstrates the generalization ability of our framework to real monocular videos.

[4]More results can be found in the appendix.

4.4 Ablation Study

Table 3: **Additional ablation of MoVE-4D on** DynObj4D**.** "GT" denotes ground-truth motion occupancy; "w/o Temporal" denotes cross-group temporal attention being removed; "w/ mean" denotes mean aggregation for CT-SLAT aggregation; "w/o d.s." denotes without downsampling; "SS4D-FT" represents fine-tuned SS4D models.

No.	Stage 1 Variants	Stage 2 Variants	CLIP↑	PSNR↑	SSIM↑	LPIPS↓	FVD↓
i	GT	Ours	0.957	24.849	0.943	0.044	282.511
ii	GT	Ours w/o Temporal	0.950	22.856	0.932	0.057	377.429
iii	GT	Ours w/ mean	0.939	21.635	0.923	0.068	486.299
iv	SS4D-FT	Ours	0.929	17.173	0.892	0.105	571.665
v	Ours w/o Temporal	Ours	0.931	17.809	0.893	0.101	639.963
vi	SS4D-FT	Ours w/o d.s.	0.924	16.825	0.888	0.106	603.710
–	SS4D-FT	SS4D-FT	0.916	16.924	0.888	0.107	683.292
–	**Ours**	**Ours**	0.935	18.525	0.900	0.090	527.888

Comparison with fine-tuned baseline. As SS4D is the baseline most closely related to our method, we fine-tune it on the training set, including its Stage-2 VAE and both generators. The results in Table 3 suggest that the performance gains of MoVE-4D are not primarily attributable to differences in the synthetic rendering distribution, but instead arise from the proposed representation and framework.

Effectiveness of motion occupancy generation. Comparing variant i with the full model in Table 3 shows that replacing predicted motion occupancy with ground-truth occupancy substantially improves generation quality, highlighting the importance of accurate structural guidance for Stage 2. In variant iv, we merge SS4D-FT's per-frame occupancy predictions within each temporal group and feed the resulting shared occupancy to the same Stage 2 model used in the full pipeline. Comparing variant iv with the full model therefore evaluates the benefit of directly predicting group-level motion occupancy in Stage 1, rather than generating per-frame occupancies and subsequently merging them.

Effectiveness of CT-SLAT. CT-SLAT is defined on shared group-level voxels, coupling temporal feature aggregation with within-group voxel sharing. To evaluate this design, variant vi uses exactly the same timestep-wise occupancy predictions as variant iv, but without occupancy merging, retaining per-frame voxel supports for latent prediction and decoding. Comparing variants iv and vi therefore controls the underlying Stage 1 occupancy predictions and evaluates the joint contribution of shared voxel support and the corresponding CT-SLAT representation in Stage 2. The results in Table 3 support the effectiveness of this joint design. We further evaluate the aggregation function by replacing the proposed MLP-based aggregator with mean aggregation over timestep features. Both variants use ground-truth motion occupancy. Comparing variant i with variant iii in Table 3 shows that learned MLP aggregation achieves higher generation quality than mean aggregation.

Effectiveness of cross-group temporal attention[5]**.** We remove cross-group temporal attention from the Stage 2 generator with ground-truth occupancy (variant ii) and from the Stage 1 generator in the full pipeline (variant v). Both variants perform worse than their corresponding references, variant i and the full model (ours), respectively. These results support the role of cross-group temporal attention in coordinating temporal groups and improving generation quality.

5 Conclusion

We introduced MoVE-4D, a two-stage feed-forward framework for efficient video-to-4D generation. Our method builds on structured 4D Gaussians and develops CT-SLAT, a compact temporal structured latent space that aggregates visual evidence across time onto motion-occupied voxels. By combining motion occupancy merging, visual feature aggregation, and temporal grouping, MoVE-4D reduces redundant per-timestep modeling while preserving the flexibility needed for complex dynamic geometry and appearance. Experiments on synthetic and real-world benchmarks demonstrate a favorable quality–efficiency trade-off with strong visual quality and temporal consistency.

[5]More ablations about the pipelines can be found in the appendix.

AI USE STATEMENT

Generative AI tools were used to assist with language editing and software implementation, including improving the grammar, clarity, and presentation of the manuscript, as well as writing and debugging portions of the research code. All AI-assisted code was reviewed, tested, and validated by the authors. Generative AI tools were not used to generate research ideas, experimental data or results, or scientific conclusions. The authors take full responsibility for all AI-assisted content and code used in this work.

ETHICS STATEMENT

This work includes a visual-preference user study in which participants evaluate generated 4D results. Participation was voluntary, and informed consent was obtained from all participants prior to participation. We did not collect personally identifiable or sensitive information, and all responses are reported only in aggregate. We have also considered potential misuse of generative 3D/4D technologies, including the creation of misleading synthetic visual content. The methods and experiments in this work are intended for research purposes, and the authors adhere to the ICLR Code of Ethics.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the proposed representation, training procedure, and implementation settings in the main paper. We also provide qualitative results and additional comparisons in the appendix to facilitate verification of the reported findings. Code and model checkpoints will be released to support reproducibility.

REFERENCES

Sherwin Bahmani, Ivan Skorokhodov, Victor Rong, Gordon Wetzstein, Leonidas Guibas, Peter Wonka, Sergey Tulyakov, Jeong Joon Park, Andrea Tagliasacchi, and David B Lindell. 4d-fy: Text-to-4d generation using hybrid score distillation sampling. In *CVPR*, 2024.

Hongyuan Chen, Xingyu Chen, Youjia Zhang, Zexiang Xu, and Anpei Chen. Motion 3-to-4: 3d motion reconstruction for 4d synthesis. *arXiv preprint*, 2026.

Yiwen Chen, Zhihao Li, Yikai Wang, Hu Zhang, Qin Li, Chi Zhang, and Guosheng Lin. Ultra3D: Efficient and high-fidelity 3d generation with part attention. *arXiv preprint arXiv:2507.17745*, 2025.

Wen-Hsuan Chu, Lei Ke, and Katerina Fragkiadaki. Dreamscene4d: Dynamic multi-object scene generation from monocular videos. *NeurIPS*, 2024.

Devikalyan Das, Christopher Wewer, Raza Yunus, Eddy Ilg, and Jan Eric Lenssen. Neural parametric gaussians for monocular non-rigid object reconstruction. In *CVPR*, 2024.

Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. Objaverse-xl: A universe of 10m+ 3d objects. *Advances in Neural Information Processing Systems*, 36:35799–35813, 2023a.

Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *CVPR*, 2023b.

Yuanxing Duan, Fangyin Wei, Qiyu Dai, Yuhang He, Wenzheng Chen, and Baoquan Chen. 4d-rotor gaussian splatting: towards efficient novel view synthesis for dynamic scenes. In *ACM SIGGRAPH*, 2024.

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 *ICML*, 2024.

Yanqin Jiang, Li Zhang, Jin Gao, Weiming Hu, and Yao Yao. Consistent4d: Consistent 360 dynamic object generation from monocular video. *arXiv preprint*, 2023.

Zeren Jiang, Chuanxia Zheng, Iro Laina, Diane Larlus, and Andrea Vedaldi. Mesh4d: 4d mesh reconstruction and tracking from monocular video. *arXiv preprint arXiv:2601.05251*, 2026.

Kai Katsumata, Duc Minh Vo, and Hideki Nakayama. A compact dynamic 3d gaussian representation for real-time dynamic view synthesis. In *ECCV*. Springer, 2024.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, George Drettakis, et al. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 2023.

Yangguang Li, Zi-Xin Zou, Zexiang Liu, Dehu Wang, Yuan Liang, Zhipeng Yu, Xingchao Liu, Yuan-Chen Guo, Ding Liang, Wanli Ouyang, et al. Triposg: High-fidelity 3d shape synthesis using large-scale rectified flow models. *IEEE TPAMI*, 2025a.

Zhibing Li, Mengchen Zhang, Tong Wu, Jing Tan, Jiaqi Wang, and Dahua Lin. Ss4d: Native 4d generative model via structured spacetime latents. *ACM TOG*, 2025b.

Huan Ling, Seung Wook Kim, Antonio Torralba, Sanja Fidler, and Karsten Kreis. Align your gaussians: Text-to-4d with dynamic 3d gaussians and composed diffusion models. In *CVPR*, 2024.

Zhicheng Lu, Xiang Guo, Le Hui, Tianrui Chen, Min Yang, Xiao Tang, Feng Zhu, and Yuchao Dai. 3d geometry-aware deformable gaussian splatting for dynamic view synthesis. In *CVPR*, 2024.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *CVPR*, 2021.

Jiawei Ren, Liang Pan, Jiaxiang Tang, Chi Zhang, Ang Cao, Gang Zeng, and Ziwei Liu. Dreamgaussian4d: Generative 4d gaussian splatting. *arXiv preprint*, 2023.

Jiawei Ren, Kevin Xie, Ashkan Mirzaei, Hanxue Liang, Xiaohui Zeng, Karsten Kreis, Ziwei Liu, Antonio Torralba, Sanja Fidler, Seung W Kim, et al. L4gm: Large 4d gaussian reconstruction model. *NeurIPS*, 2024.

Remy Sabathier, David Novotny, Niloy J Mitra, and Tom Monnier. Actionmesh: Animated 3d mesh generation with temporal 3d diffusion. *arXiv preprint*, 2026.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. LGM: Large multi-view gaussian model for high-resolution 3d content creation. In *ECCV*, 2024.

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.

Feng Wang, Sinan Tan, Xinghang Li, Zeyue Tian, Yafei Song, and Huaping Liu. Mixed neural voxels for fast multi-view video synthesis. In *ICCV*, 2023.

Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4D gaussian splatting for real-time dynamic scene rendering. In *CVPR*, 2024.

Zijie Wu, Chaohui Yu, Fan Wang, and Xiang Bai. Animateanymesh: A feed-forward 4d foundation model for text-driven universal mesh animation. In *ICCV*, 2025.

Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3d latents for scalable and versatile 3d generation. In *CVPR*, 2025.

Zeyu Yang, Hongye Yang, Zijie Pan, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. *ICLR*, 2024.

Chun-Han Yao, Yiming Xie, Vikram Voleti, Huaizu Jiang, and Varun Jampani. Sv4d 2.0: Enhancing spatio-temporal consistency in multi-view video diffusion for high-quality 4d generation. In *ICCV*, 2025.

Jiraphon Yenphraphai, Ashkan Mirzaei, Jianqi Chen, Jiaxu Zou, Sergey Tulyakov, Raymond A Yeh, Peter Wonka, and Chaoyang Wang. Shapegen4d: Towards high quality 4d shape generation from videos. *arXiv preprint arXiv:2510.06208*, 2025.

Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3Dshape2VecSet: A 3D shape representation for neural fields and generative diffusion models. *ACM TOG*, 2023.

Bowen Zhang, Sicheng Xu, Chuxin Wang, Jiaolong Yang, Feng Zhao, Dong Chen, and Baining Guo. Gaussian variation field diffusion for high-fidelity video-to-4d synthesis. In *ICCV*, 2025a.

Haiyu Zhang, Xinyuan Chen, Yaohui Wang, Xihui Liu, Yunhong Wang, and Yu Qiao. 4diffusion: Multi-view video diffusion model for 4d generation. *NeurIPS*, 2024.

Xinjie Zhang, Zhening Liu, Yifan Zhang, Xingtong Ge, Dailan He, Tongda Xu, Yan Wang, Zehong Lin, Shuicheng Yan, and Jun Zhang. Mega: Memory-efficient 4d gaussian splatting for dynamic scenes. In *ICCV*, pp. 27828–27838, 2025b.

Zibo Zhao, Zeqiang Lai, Qingxiang Lin, Yunfei Zhao, Haolin Liu, Shuhui Yang, Yifei Feng, Mingxin Yang, Sheng Zhang, Xianghui Yang, et al. Hunyuan3d 2.0: Scaling diffusion models for high resolution textured 3d assets generation. *arXiv preprint arXiv:2501.12202*, 2025.

A Additional Ablations

Effectiveness of the 4D representation generation. We compare our method with another way to improve generation efficiency: directly increasing the downsampling rate in the model. To reach the same downsampling rate as ours to both stages of SS4D, we add 1D CNN downsampling and upsampling layers along the temporal dimension, synchronized with the original spatial downsampling and upsampling stages in the voxel sequence VAE. For dynamic appearance generation, SS4D originally applies a $2\times$ temporal downsampling before the DiT blocks. We further insert additional sparse 1D CNN downsampling and the corresponding upsampling layers, increasing the temporal downsampling ratio. As shown in Table A3, we compare it with our design. The performance of SS4D drops noticeably under this setting and remains inferior to ours. This suggests that simply increasing the downsampling rate removes useful spatial-temporal information indiscriminately, whereas our structured 4D Gaussian representation provides motion-occupied support and time-aware primitives that preserve shared geometry and appearance while reducing redundant content.

Validation on different τ. We evaluate three motion occupancy compression settings: no compression, $\tau = 8$, and $\tau = 12$. The no-compression variant keeps timestep-wise occupancy without temporal merging, corresponding to SS4D-style first-stage generation with a 4DGS-based representation. As shown in Table A3, it performs slightly worse than SS4D, suggesting that simply adopting 4DGS is insufficient to exploit its temporal modeling ability. With highly fine-grained timestep-wise occupancy, 4D Gaussians have an extremely short average temporal scale for the object (0.0001 vs. 0.0512 under $\tau = 8$), making them behave similarly to independent 3D Gaussians. In contrast, motion occupancy merging with $\tau = 8$ increases the temporal lifespan of 4D Gaussians, enabling stronger temporal inheritance of appearance states and more effective redundancy reduction. Further increasing the window size to $\tau = 12$ degrades performance compared with $\tau = 8$. This suggests that excessive temporal aggregation can weaken fine-grained temporal modeling. Overall, these results validate temporal grouping as an effective balance between redundancy reduction and dynamic detail preservation.

Validation of temporal attention in the VAE. Temporal attention is also present in the CT-SLAT VAE. Removing it lowers reconstruction PSNR from 30.33 to 22.70 and increases FVD from 157.36 to 515.11 (Table A1).

Validation of Gaussians per voxel. Table A2 compares 8, 32, and 64 Gaussians per voxel. Using 8 reduces reconstruction quality, while increasing from 32 to 64 gives no improvement on the reported metrics. We use 32 to balance capacity and cost.

Table A1: **Temporal attention in the CT-SLAT VAE.** Reconstruction on DynObj4D, averaged over tested views. Cell colors indicate 1st and 2nd per column; ties share a color.

VAE setting	PSNR↑	FVD↓
w/o temporal attention	22.70	515.11
w/ temporal attention	30.33	157.36

Table A2: **Number of 4D Gaussians per voxel.** VAE reconstruction on DynObj4D. Cell colors indicate 1st and 2nd per column; ties share a color.

Gaussians per voxel	PSNR↑	FVD↓
8	30.07	163.75
32 (default)	30.33	157.36
64	30.29	159.28

Table A3: **Ablation studies on** DYNOBJ4D**.** We compare different temporal downsampling (d.s.) ratios. Cell colors indicate 1st, 2nd, and 3rd per column.

Method	CLIP↑	PSNR↑	SSIM↑	LPIPS↓	FVD↓
SS4D	0.926	16.920	0.890	0.105	593.568
SS4D (8× d.s.)	0.913	16.836	0.881	0.113	775.534
Ours (w/o d.s.)	0.924	16.825	0.888	0.106	603.710
Ours (12× d.s.)	0.919	18.267	0.898	0.097	642.058
Ours (8× d.s.)	0.935	18.525	0.900	0.090	527.888

B TEMPORAL-GROUP BOUNDARY DIAGNOSTICS

For the sequences considered here, Gaussians from all temporal groups are merged into a unified representation and rendered jointly. The outputs are not independently rendered clips joined at hard boundaries. Temporal attention in the generators and VAE provides additional cross-group modeling.

Tables A4 and A5 report reconstruction scores for individual frames and their changes on the DYNOBJ4D test set, averaged over four views. Under the reported frame indexing, group boundaries occur at $7 \to 8$ and $15 \to 16$ for $\tau = 8$. The listed boundary transitions show no pronounced drop in these reconstruction metrics relative to the neighboring transitions.

Table A4: **Frame-wise reconstruction near group boundaries.** Bold frame indices identify frames immediately before and after each boundary.

Metric	6	**7**	**8**	9	14	**15**	**16**	17
PSNR↑	18.463	18.553	18.554	18.516	18.497	18.511	18.540	18.478
SSIM↑	0.8997	0.9004	0.9006	0.9004	0.8998	0.9000	0.9007	0.9003
LPIPS↓	0.0899	0.0889	0.0882	0.0883	0.0894	0.0892	0.0887	0.0892

Table A5: **Consecutive-frame changes in reconstruction metrics.** Differences are later minus earlier frame; bold headers identify cross-boundary transitions.

Metric	$6 \to 7$	$\mathbf{7 \to 8}$	$8 \to 9$	$14 \to 15$	$\mathbf{15 \to 16}$	$16 \to 17$
ΔPSNR	+0.090	+0.001	−0.038	+0.013	+0.029	−0.062
ΔSSIM	+0.0006	+0.0002	−0.0002	+0.0002	+0.0007	−0.0004
ΔLPIPS	−0.0011	−0.0007	+0.0002	−0.0002	−0.0005	+0.0004

C ADDITIONAL QUALITATIVE RESULTS

Figures A1–A3 provide additional comparisons on DYNOBJ4D. Figure A4 shows real-world examples from DAVIS4DTEST. The comparisons include Gaussian-based 4D generation frameworks like L4GM (Ren et al., 2024), GVF (Zhang et al., 2025a), SS4D (Li et al., 2025b), and mesh-based 4D generation frameworks like Mesh4D (Jiang et al., 2026) and Motion-3-to-4 (Chen et al., 2026). Static frames illustrate object shape and appearance, while temporal flickering and motion artifacts require assessment through video comparisons.

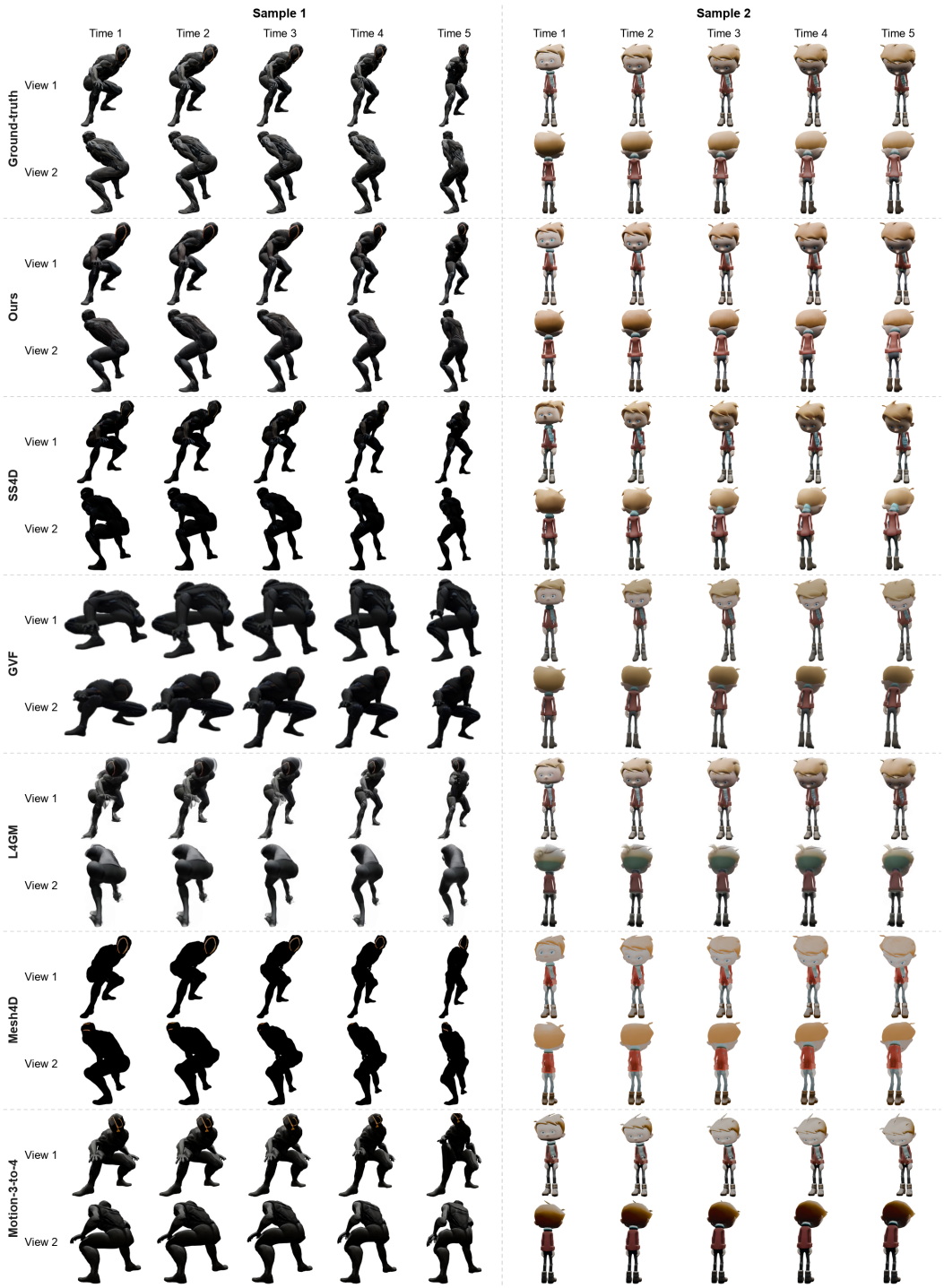


Figure A1: Additional qualitative comparisons on DYNOBJ4D.

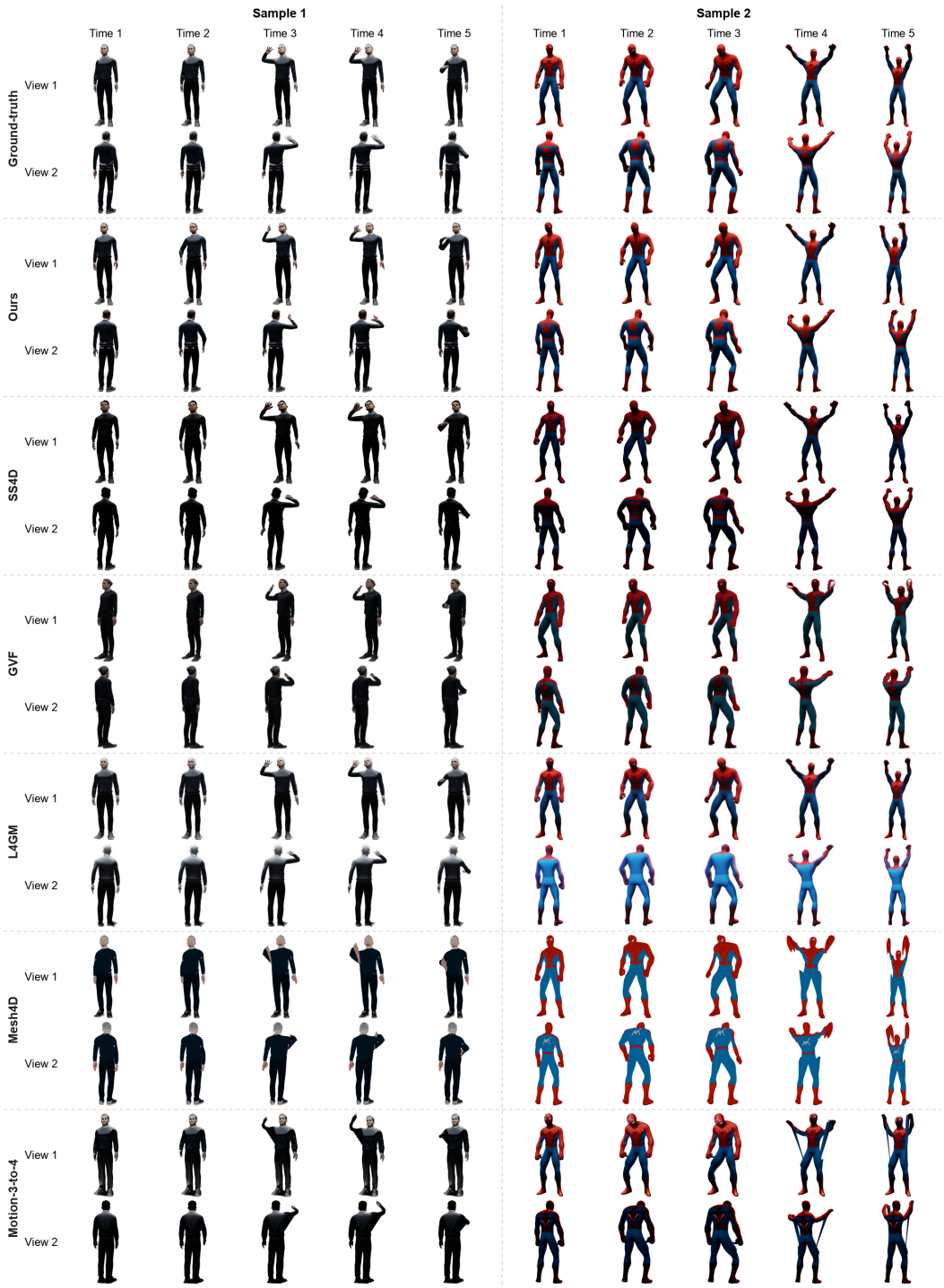


Figure A2: Additional qualitative comparisons on DYNOBJ4D.

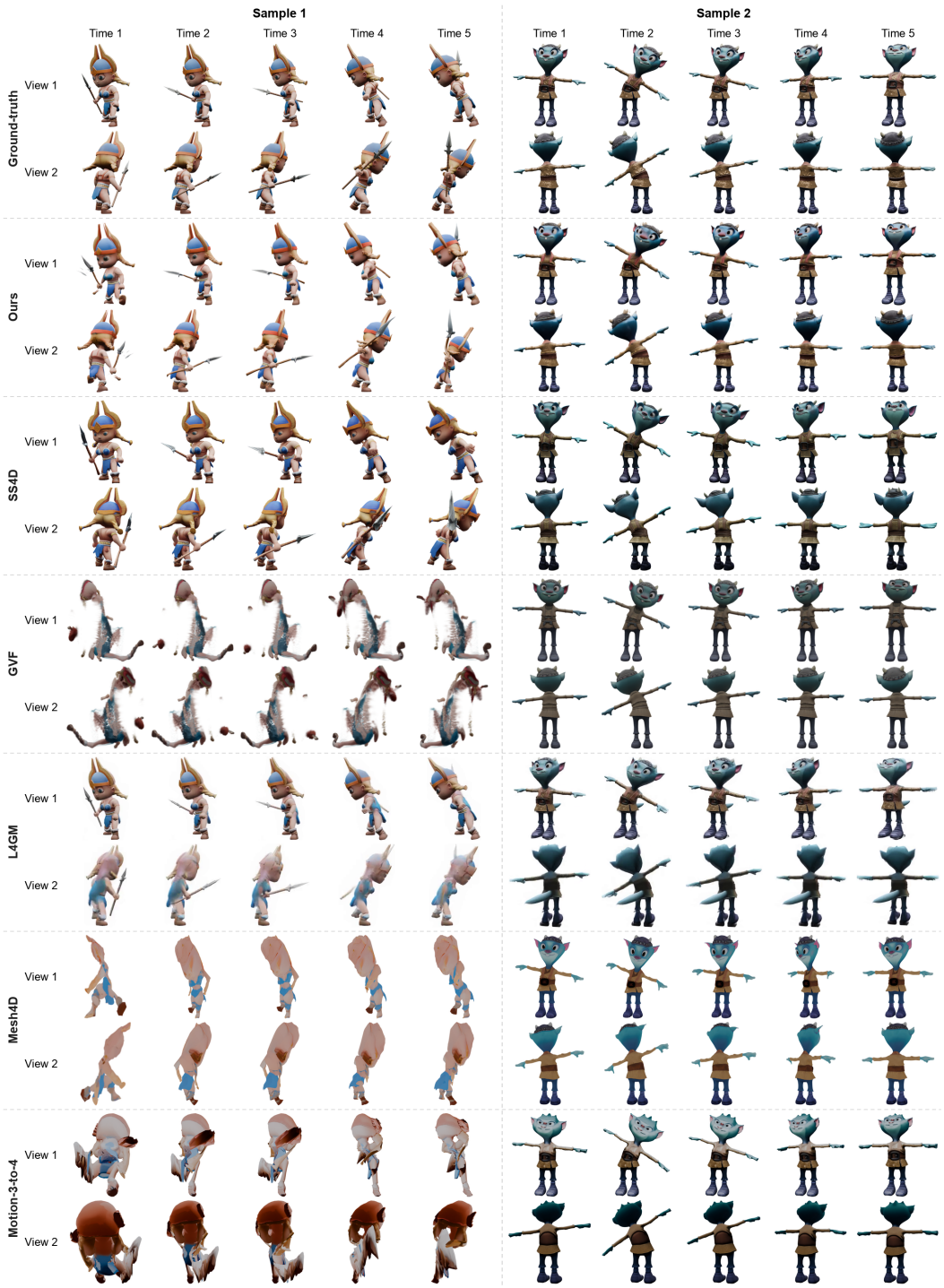


Figure A3: Additional qualitative comparisons on DynObj4D.



Figure A4: Additional qualitative comparisons on DAVIS4DTEST.

D LIMITATIONS AND FUTURE WORK

Our method may degrade on challenging out-of-distribution cases, particularly those involving very fast, complex motion. Typical failures include blurred dynamic regions and deformation artifacts, as shown in Fig. A5. These issues mainly arise from the lack of large-scale real-world 4D supervision, which remains a common challenge in video-to-4D generation. Future work will improve OOD robustness by introducing augmented real-world training data obtained by monocular depth estimation and point-cloud completion, using more diverse synthetic rendering, and employing advanced 4DGS renderers.

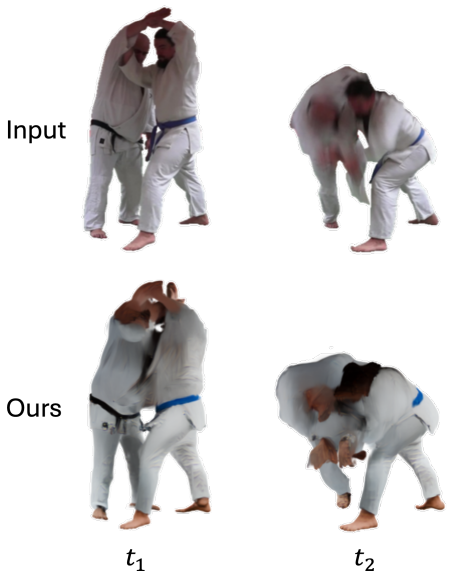


Figure A5: Failure case.