FLUXSPLAT: FEED-FORWARD GAUSSIAN SPLATTING FOR STREAMABLE DYNAMIC VIEW SYNTHESIS FROM MONOCULAR VIDEOS

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper addresses streamable dynamic view synthesis (DVS) from continuous monocular video streams using feed-forward dynamic Gaussian Splatting. Existing methods either process limited-length sequences as a whole or rely on adjacent frames, limiting spatial and temporal context and causing flickering in long streams. Our goal is to enable streamable DVS over long streams with limited resource consumption. We propose FluxSplat, a feed-forward framework that can incorporate historical information through a History-aware Depth Prediction Head for temporally consistent Gaussian prediction and enables compact Gaussian representation with a Compact Gaussian Adapter that allows high-resolution rendering with fewer hole artifacts. To efficiently handle long streams, we introduce Motion-aware Farthest View Sampling that selects informative frames to reduce both spatial and temporal redundancy. Experiments on benchmark datasets and in-the-wild videos show that FluxSplat achieves the state-of-the-art performance.

1 INTRODUCTION

Continuously representing dynamic scenes from monocular videos and synthesizing their evolving appearance from novel viewpoints, referred to as streamable dynamic view synthesis (DVS), is an important task in computer vision, with applications in embodied AI and Virtual Reality.

Feed-forward dynamic Gaussian Splatting (DGS) models enable fast DVS of dynamic scenes, but existing approaches struggle to scale to continuously growing monocular video streams. Most methods (Lin et al., 2026a; Xu et al., 2025; Kim et al., 2026; Yang et al., 2026) process an entire sequence jointly in an offline fashion, causing computation, memory, and the number of Gaussians to grow with the stream length. Streamable methods (Wu et al., 2026; Hur et al., 2026) improve scalability by processing adjacent views, but limited observations hinder the modeling of the relationship between scene coverage and motion over time which leads to temporal flickering and inconsistencies of the rendered video. Others (Balice et al., 2026; Chen et al., 2026; Yang et al., 2025) maintain richer multi-view context, but rely on multi-camera setups rather than a monocular video.

In this work, we aim to enable feed-forward DVS for continuously growing monocular streams while maintaining sufficient spatial and temporal context with limited resource consumption. To this end, we introduce **FluxSplat**, a feed-forward framework that effectively leverages both historical sequence context and each incoming frame to reconstruct dynamic scenes with 3D Gaussians enabling improved temporally consistent dynamic-view video rendering.

Our framework is built on pretrained models (Wang et al., 2025a; Lin et al., 2026a) and initially maps each pixel to a 3D Gaussian primitive, whose position is determined by the predicted depth, while explicitly modeling pixel-wise motion. To enable temporally consistent online Gaussian prediction, we introduce a new History-aware Depth Prediction Head to maintain a latent state for each timestep. This design allows historical information to propagate across the stream and improve temporally consistent Gaussian predictions. To construct a more compact Gaussian representation, we further propose a Compact Gaussian Adapter that enables neighboring Gaussians to coordinate their spatial coverage, reducing the number of required primitives, improving scale prediction accuracy, and supporting high-resolution rendering with fewer hole artifacts. As new frames arrive, FluxSplat incrementally update a time-varying Gaussian representation of dynamic scenes.

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.

However, as new frames continuously arrive, the monocular video stream grows increasingly long, making it progressively more computationally expensive to process the entire increasing long sequence. We note that a compact subset of frames are always sufficient to model the dynamic scene captured by the dense long sequence. We therefore propose **Motion-aware Farthest View Sampling (Motion-FVS)** to select a subset of frames, which captures spatial and temporal diversity of the scene. We are tackling *in-distribution* streamable DVS for which we can identify the observed frame with the closest camera pose to the novel-view pose and focus on modeling the dynamics toward the query time given a query pose and time. With such Motion-FVS, FluxSplat can achieve streamable DVS maintaining a constant inference memory for streamable DVS which enables its capability to handle long sequences efficiently. Our main contributions are:

1. We introduce **FluxSplat** – a feed-forward framework for streamable DVS, featuring a History-aware Depth Prediction Head for temporally consistent Gaussian predictions. We further propose a Compact Gaussian Adapter to predict compact Gaussians with more accurate scales, reducing the number of Gaussians and hole artifacts in high-resolution rendering.
2. We propose a simple yet training-free **Motion-Aware Farthest View Sampling** strategy that selects a compact and informative set of frames, enabling streaming DVS with constant inference memory over long video streams.

Extensive experiments on three challenging DVS datasets (Gao et al., 2022; Stearns et al., 2024; Park et al., 2021a;b) and our in-the-wild benchmark (Pont-Tuset et al., 2017; Rockwell et al., 2025) demonstrate FluxSplat achieves SOTA performance for streamable DVS from monocular streams.

2 Related Works

Dynamic Scene Reconstruction. 3D reconstruction has evolved from optimization-based multi-view geometry and SfM (Schonberger & Frahm, 2016) to feed-forward models that directly infer geometry and camera poses (Wang et al., 2024; Leroy et al., 2024; Wang et al., 2025a), with recent work extending to streaming videos (Zhuo et al., 2026; Yuan et al., 2026). 4D reconstruction has followed a similar shift, from optimization-based (Li et al., 2025; Huang et al., 2025) to feed-forward methods (Zhang et al., 2025; Wang et al., 2025d; Lin et al., 2026b; Zhou et al., 2026). Point-tracking-based methods (Xiao et al., 2025; Feng et al., 2025; Karhade et al., 2026; Zhang et al., 2026; Jeon et al., 2026) further model spatiotemporal point trajectories. All above mainly target explicit 4D reconstruction, while we focus on streamable DVS from monocular streams.

Dynamic Gaussian Splatting. DGS has been widely extended to dynamic scenes (Yang et al., 2024; Huang et al., 2024a), with explicit motion and geometric priors further improving reconstruction (Wang et al., 2025c; Lei et al., 2025; Stearns et al., 2024; Park et al., 2025). However, these methods typically rely on per-scene optimization, incurring much training costs. Feed-forward DGS removes this dependency and enables rapid prediction of dynamic Gaussians. Recent works study offline DVS from monocular videos (Liang et al., 2025; Lin et al., 2026a; Shen et al., 2025; Xu et al., 2025; Lin et al., 2025; Yan et al., 2026; Ma et al., 2026), with some incorporating video diffusion for improved quality (Kim et al., 2026; Yang et al., 2026). For streaming, StreamSplat (Wu et al., 2026) and UFO-4D (Hur et al., 2026) use adjacent-view inputs, while others extend feed-forward DGS to multi-view streams (Wang et al., 2025e; Balice et al., 2026). Specialized methods further target driving scenes (Chen et al., 2026; Wang et al., 2025b; Yang et al., 2025) or dynamic objects (Ma et al., 2025; Ren et al., 2024). Existing methods assume full videos, rely on adjacent views, or target specific scenarios, whereas our FluxSplat enables streamable DVS from continuous monocular streams with a bounded input budget while exploiting spatial and temporal redundancy.

3 Preliminary and Formulation

We follow the formulation of MoVieS (Lin et al., 2026a), an offline feed-forward pixel-wise dynamic Gaussian Splatting model with an architecture similar to that shown in Fig. 1. Given a posed monocular video $\mathcal{V} = \{I_t, \mathbf{P}_t, \mathbf{K}_t\}_{t=1}^{L}$, where $I_t \in \mathbb{R}^{3\times H\times W}$, $\mathbf{P}_t \in SE(3)$, and $\mathbf{K}_t \in \mathbb{R}^{3\times 3}$ denote the frame, camera pose, and intrinsics at timestep $t \in [0, 1]$, respectively. A feedforward network $\mathcal{F}_\theta$ predicts per-frame depth maps $\mathcal{D}$, a 3D motion field $\mathcal{M}$ (time-varying per-depth-point positional deformation between different timesteps), attributes of canonical 3D Gaussians $\mathcal{G}^c$, and corresponding deformation fields $\Delta\mathcal{G}$ between different timesteps: $\mathcal{F}_\theta(\mathcal{V}) = (\mathcal{D}, \mathcal{M}, \mathcal{G}^c, \Delta\mathcal{G})$. The canonical rep-

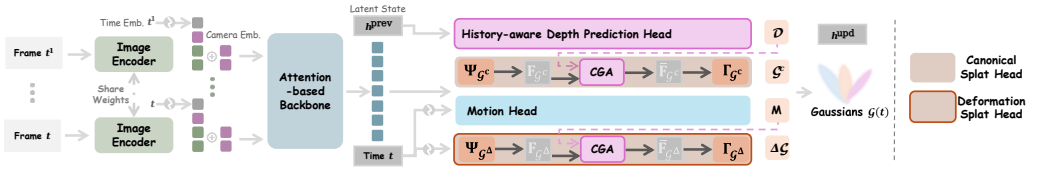

Figure 1: FluxSplat comprises an encoder, backbone, and four heads for predicting depth $\mathcal{D}$, motion field $\mathcal{M}$, canonical Gaussian attributes $\mathcal{G}^c$, and deformation field $\Delta\mathcal{G}$. Given frames with camera and time embeddings, the Motion and Deform. Splat Heads are time-conditioned, while our depth head maintains a latent state for each timestep. Compact Gaussian Adapter (CGA) is integrated into both Splat Heads, and their outputs are aggregated into time-varying compact Gaussians $\mathcal{G}(t)$.

resentation $\mathcal{G}^c = \{g_i^c\}_{i=1}^N$ consists of Gaussian primitives $g_i^c = (\mathbf{x}_i, \mathbf{c}_i, \mathbf{o}_i, \mathbf{r}_i, \mathbf{s}_i)$, where $\mathbf{x}_i \in \mathbb{R}^3$, $\mathbf{c}_i \in \mathbb{R}^3$, $\mathbf{o}_i \in \mathbb{R}$, $\mathbf{r}_i \in \mathbb{R}^4$, and $\mathbf{s}_i \in \mathbb{R}^3$ denote position, color, opacity, rotation, and scale, respectively. The positions $\mathbf{x}_i$ are lifted from the predicted depth $\mathcal{D}$. Temporal dynamics are represented by per-Gaussian deformation fields $\Delta g_i(t) = (\Delta\mathbf{x}_i(t), \Delta\mathbf{c}_i(t), \Delta\mathbf{o}_i(t), \Delta\mathbf{r}_i(t), \Delta\mathbf{s}_i(t))$, where the positional deformation $\Delta\mathbf{x}_i(t)$ is derived from the motion field $\mathcal{M}$. The scene representation at timestep t is given by: $\mathcal{G}(t) = \{g_i(t)\}_{i=1}^N$, where $g_i(t) = g_i^c +^1 \Delta g_i(t)$, and can be used for DVS.

4 METHOD

We present **FluxSplat**, a framework to transform online video streams into dynamic 3D Gaussians with constant inference memory, enabling streamable DVS. The overall architecture is shown in Fig. 1. Specifically, given a video stream $\mathcal{V}_{t_\tau} = \{I_t, \mathbf{P}_t, \mathbf{K}_t\}_{t=1}^{t_\tau}$, where t_τ denotes the current timestep, streamable DVS supports to synthesize images at any given time step t_q where $t_q \in \{1, \ldots, t_\tau\}$ and from any *in-distribution* camera view $\mathbf{P}_q$. Below, we first introduce History-aware Depth Prediction Head (Section 4.1) and Compact Gaussian Adapter (Section 4.2) in our FluxSplat. We then present Motion-FVS (Section 4.3), a simple yet training-free strategy for sampling the compact frame set. Finally, we describe training and inference in Section 4.4.

4.1 HISTORY-AWARE DEPTH PREDICTION HEAD

To handle the growing number of frames in a video stream, we dynamically sample a set of frames as new frames arrive, enabling 3D scene reconstruction with constant memory consumption. We defer the details of our proposed sampling strategy to Section 4.3. Due to the dynamic sampling, the reconstruction shows varying spatial and temporal contexts and inconsistent temporal Gaussian position predictions. To address these issues, we introduce a History-aware Depth Prediction Head (HDP Head) that augments the conventional dense prediction (DPT) head with a GRU-based (Chung et al., 2014) latent state at each timestep. This state helps historical information propagate across the stream, reducing prediction fluctuations over time. We show the procedure in Fig. 2(c). For each frame, given its feature $\mathbf{F}_{\text{attn}}$ from the attention-based backbone, the HDP Head updates the previous latent states $\mathbf{h}^{\text{prev}}$ and predicts the corresponding depth $\mathcal{D}$:

$$\tilde{\mathbf{F}}_{\mathcal{D}} = \Psi_{\mathcal{D}}(\mathbf{F}_{\text{attn}}), \quad \mathbf{h}^{\text{upd}} = \text{GRU}(\tilde{\mathbf{F}}_{\mathcal{D}}, \mathbf{h}^{\text{prev}}), \quad \mathcal{D} = \Gamma_{\mathcal{D}}(\tilde{\mathbf{F}}_{\mathcal{D}}, \mathbf{h}^{\text{upd}}), \tag{1}$$

where $\Psi_{\mathcal{D}}$ denotes the multi-scale feature refinement and aggregation module in the conventional DPT Head, and $\Gamma_{\mathcal{D}}$ denotes the rest prediction module. This allows Gaussian position predictions to be temporally consistent across different queries.

4.2 COMPACT GAUSSIAN ADAPTER

Existing methods (Lin et al., 2026a; Yang et al., 2026) typically adopt pixel-wise Gaussian prediction, which is convenient but can introduce redundancy and inaccurate scales, leading to increasing memory costs as the stream grows or severe hole artifacts in high-resolution (HR) rendering. Since neighboring pixels often share similar textures and can be represented by shared Gaussians, we seek a compact representation with accurate scales.

[1] In Euclidean space, "+" denotes vector addition; for rotations in $\text{SO}(3)$, it denotes rotation composition.

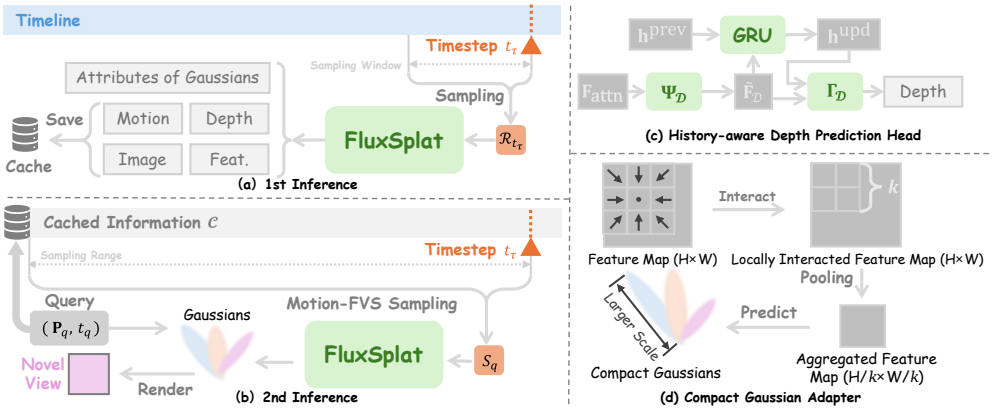

Figure 2: **Left:** Inference overview. **(a)** 1st inference: When a new frame arrives, we sample a set of frames $\mathcal{R}_{t_\tau}$ within the allowed window and store the predicted motion, depth, image, and feat. (frame feature and latent state) in the cache $\mathcal{C}$. **(b)** 2nd inference: given a query $(\mathbf{P}_\mathrm{q}, t_\mathrm{q})$ for a target novel view, we perform Motion-FVS over the cache $\mathcal{C}$ to select informative frames $\mathcal{S}_q$, which are used to predict Gaussians for queried view rendering. **Right:** **(c)** History-aware Depth Prediction Head, we argument the conventional depth DPT with a GRU. **(d)** Compact Gaussian Adapter aggregates local neighborhood features to predict compact Gaussians with larger scales.

To this end, we propose the Compact Gaussian Adapter (CGA), as shown in Fig. 2(d). CGA exploits the natural correlation between neighboring pixels through a 3×3 local neighborhood, capturing local dependencies and aggregate features to reduce the number of Gaussians. Specifically, for local feature interaction, CGA predicts a local interaction map $\mathbf{A} \in \mathbb{R}^{9\times H\times W}$: $\mathbf{A} = \Lambda([\mathbf{F}_\mathcal{G}; \mathbf{Y}_\mathcal{G}])$, from the pixel-wise Gaussian attribute features $\mathbf{F}_\mathcal{G}$, positions or their deformations $\mathbf{Y}_\mathcal{G}$. $\mathbf{F}_\mathcal{G}$ is obtained from the multi-scale feature refinement and aggregation module $\Psi_\mathcal{G}$ in the two Splat heads, and $\mathbf{Y}_\mathcal{G}$ is derived from depth and motion heads. For each pixel p, given its 3×3 neighborhood $\mathcal{N}(p)$ and the predicted interaction weight $\mathbf{A}_{p,r}$, where $r \in \mathcal{N}(p)$, we update the pixel-wise Gaussian attribute feature through local interaction:

$$\hat{\mathbf{X}}_\mathcal{G}(p) = \frac{\sum_{r\in\mathcal{N}(p)} \mathbf{A}_{p,r}\mathbf{X}_\mathcal{G}(r)}{\sum_{r\in\mathcal{N}(p)} \mathbf{A}_{p,r}}. \tag{2}$$

where $\mathbf{X}_\mathcal{G}$ denotes either $\mathbf{F}_\mathcal{G}$ or $\mathbf{Y}_\mathcal{G}$. After local interaction, we apply average pooling to $\hat{\mathbf{X}}_\mathcal{G}$ using a kernel size k and stride of k, aggregating information within each neighborhood while reducing its spatial resolution from $H \times W$ to $\frac{H}{k} \times \frac{W}{k}$: $\bar{\mathbf{X}}_\mathcal{G} = \mathrm{Pool}_k(\hat{\mathbf{X}}_\mathcal{G})$. The pooled feature $\bar{\mathbf{F}}_\mathcal{G}$ is fed into the subsequent prediction module $\Gamma_\mathcal{G}$ to predict Gaussian attributes: $\Gamma_\mathcal{G}(\bar{\mathbf{F}}_\mathcal{G})$, while the pooled $\bar{\mathbf{Y}}_\mathcal{G}$ provides the corresponding geometric information. The overall procedure of CGA is shown in Fig. 1. The kernel k directly controls the number of predicted Gaussians, reducing the per-pixel count by k^2. CGA consequently yields compact Gaussians with accurate scales, enabling each Gaussian to cover a larger spatial region and reducing hole artifacts without requiring HR supervision.

4.3 Motion-aware Farthest View Sampling

As the video stream $\mathcal{V}_{t_\tau}$ grows, processing the entire stream becomes progressively more expensive. For any query $(\mathbf{P}_q, t_q)$, however, only a subset of the observed frames is needed to model the scene evolution relevant to the query. In particular, since we focus on the scene dynamics observed at $\mathbf{P}_q$, we can identify the observed frame with the camera pose closest to $\mathbf{P}_q$ and model the dynamics from this frame toward t_q. Even along this trajectory, adjacent frames contain spatial and temporal redundancy, allowing the dynamics to be captured by a compact set. Thus, we seek a compact set $\mathcal{S}_q$ of K frames that captures the spatial and temporal evolution of the scene along this query-relevant trajectory. We introduce Motion-aware Farthest View Sampling (Motion-FVS), a two-stage strategy that first filters out frames irrelevant to the query and then selects informative frames from the remaining candidates using the cache $\mathcal{C}$, where $\mathcal{C}_\mathrm{sem}$ stores per-frame semantic features from the image encoder DINOv2 (Oquab et al., 2024), and $\mathcal{C}_\mathrm{motion}$ stores adjacent-frame motion. Specifically,

Relevance Filtering constructs a query-relevant set $\mathcal{U}$ based on content consistency, while *Motion-Guided Sampling* selects a compact set $\mathcal{S}_q$ by jointly considering spatial and temporal diversity.

Relevance Filtering. We first construct a candidate set $\mathcal{U}$ for the query $(\mathbf{P}_q, t_q)$. We identify the frame $I_{t_\mathbf{P}}$ whose camera pose is closest to $\mathbf{P}_q$ by minimizing the Euclidean distance between camera centers. For dense streaming poses, $I_{t_\mathbf{P}}$ provides a strong spatial anchor and view coverage for $\mathbf{P}_q$. However, the frame I_{t_q} may not contain the content visible from $\mathbf{P}_q$. We therefore traverse frames temporally from $I_{t_\mathbf{P}}$ toward I_{t_q}, to identify $I_{t_\mathbf{T}}$, the frame closest to I_{t_q} that still contains relevant content. Inspired by DINO-Tracker (Tumanyan et al., 2024), we measure content consistency by computing the cosine similarity between the semantic features $\mathcal{C}_{\text{sem}}(I_{t_\mathbf{P}})$ and $\mathcal{C}_{\text{sem}}\ (I_u)$ of each frame encountered between $t_\mathbf{P}$ and t_q. We retain frames whose similarity exceeds the threshold η_{sem} to produce the initial set $\mathcal{S}_q = \{I_{t_\mathbf{P}}, I_{t_\mathbf{T}}\}$ and a candidate set $\mathcal{U} = \{I_u\} \setminus \mathcal{S}_q$, where I_u denotes a retained candidate frame.

Motion-Guided Sampling. We then sample informative frames from $\mathcal{U}$ by considering spatial and temporal diversity to form $\mathcal{S}_q$. For *spatial diversity*, we seek frames in $\mathcal{S}_q$ that cover a broader scene to reduce spatial redundancy. For each candidate $I_u \in \mathcal{U}$, we first measure its spatial diversity with respect to the current selected set $\mathcal{S}_q$ as $d_{\text{s}}(I_u) = \min_{I_s \in \mathcal{S}_q} \|\mathbf{c}(I_u) - \mathbf{c}(I_s)\|_2$, where $\mathbf{c}(I)$ denotes the camera center of frame I.

Algorithm 1 Motion-FVS

Require: Video stream $\mathcal{V}_{t_\tau}$, cached semantic feature $\mathcal{C}_{\text{sem}}$ and motion $\mathcal{C}_{\text{motion}}$, query pose $\mathbf{P}_q$, query timestep t_q, target number K, weight λ_{ts}
Ensure: Sampled frame set $\mathcal{S}_q$

$(\mathcal{S}_q, \mathcal{U}) \leftarrow \text{RelevanceFilter}(\mathbf{P}_q, t_q, \mathcal{C}_{\text{sem}}, \mathcal{V}_{t_\tau})$
while $|\mathcal{S}_q| < K$ and $\mathcal{U} \neq \emptyset$ **do**
  for all $I_u \in \mathcal{U}$ **do**
    $d_{\text{s}}(I_u) \leftarrow \min_{I_s \in \mathcal{S}_q} \|\mathbf{c}(I_u) - \mathbf{c}(I_s)\|_2$
    $\mathcal{S}_{\text{tmp}} \leftarrow \text{TemporalSort}(\mathcal{S}_q \cup \{I_u\})$
    $\{v_m\} \leftarrow \text{Motion}(\mathcal{S}_{\text{tmp}}, \mathcal{C}_{\text{motion}})$
    $d_{\text{m}}(I_u) \leftarrow -\text{Var}(\{v_m\})$
    $\text{score}(I_u) \leftarrow d_{\text{s}}(I_u) + \lambda_{\text{ts}} d_{\text{m}}(I_u)$
  end for
  $I_u^* \leftarrow \arg\max_{I_u \in \mathcal{U}} \text{score}(I_u)$
  $\mathcal{S}_q \leftarrow \mathcal{S}_q \cup \{I_u^*\}; \mathcal{U} \leftarrow \mathcal{U} \setminus \{I_u^*\}$
end while
return $\mathcal{S}_q$

For *temporal diversity*, we seek similar motion between temporally consecutive frames in $\mathcal{S}_q$ to maintain uniform information gain, thereby reducing temporal redundancy. To measure it, we temporarily add I_u to $\mathcal{S}_q$ and temporally sort the resulting set to obtain $\mathcal{S}_{\text{tmp}}$. We then partition $\mathcal{S}_{\text{tmp}}$ into consecutive temporal intervals. Since $\mathcal{C}_{\text{motion}}$ stores the motion between every pair of adjacent frames, for any two temporally consecutive frames $I_a, I_{a+b} \in \mathcal{S}_{\text{tmp}}$, the motion $v_m(I_a, I_{a+b})$ over the interval from I_a to I_{a+b} can be computed by summing mean motion magnitudes (see Eq. 6 in the appendix) of all adjacent frame pairs between I_a and I_{a+b}. We use the variance of interval-wise motion as a proxy for temporal redundancy: smaller motion indicates less information, and vice versa. We thus favor a more uniform motion distribution, where each interval contributes similar information. We define $d_{\text{m}}(I_u) = -\text{Var}\left(\{v_m(I_a, I_{a+b})\}\right)$, where $\{v_m(I_a, I_{a+b})\}$ denotes the motion of all temporally consecutive frame pairs in $\mathcal{S}_{\text{tmp}}$. A higher $d_{\text{m}}(I_u)$ indicates a more uniform motion distribution.

The final $\text{score}(I_u) = d_{\text{s}}(I_u) + \lambda_{\text{ts}} d_{\text{m}}(I_u)$, where λ_{ts} weights temporal diversity. We greedily select the highest-score candidate I_u^* until K frames are obtained (Alg. 1), yielding a compact $\mathcal{S}_q$ that captures spatial structure and temporal dynamics while keeping inference memory costs independent of video length. Also, camera poses, scene motion, and frame features can come from FluxSplat or off-the-shelf models (Lin et al., 2026b; Teed & Deng, 2020), making Motion-FVS training-free.

4.4 Training and Inference

Training. We train FluxSplat on incoming streams mainly following the objectives from (Lin et al., 2026a), including depth loss $\mathcal{L}_{\text{depth}}$, motion loss $\mathcal{L}_{\text{motion}}$, and rendering loss $\mathcal{L}_{\text{img}}$. At each timestep t_τ, given the observed stream $\mathcal{V}_{t_\tau}$, we sample a frame set $\mathcal{R}_{t_\tau}$ in a temporal window, which always includes the current and preceding frames. FluxSplat predicts per-frame depth and semantic feature, inter-frame motion, and Gaussian attributes, which are aggregated to render input frames and supervised using above objectives. The predictions except Gaussian attributes, together with the rendered frames, are cached in cache $\mathcal{C}$ and fixed once stored.

Since sampled sets at different timesteps partially overlap, the same frame can be predicted under different contexts, leading to inconsistent depth and appearance predictions. Ground-truth super-

Table 1: Normal resolution quantitative comparison on streamable DVS. 1st, 2nd, 3rd.

Method	Nerfies & HyperNeRF			Dynamic NVIDIA			DyCheck		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
C4G (Kim et al., 2026)	14.33	0.40	0.67	16.23	**0.56**	0.38	11.20	0.24	0.83
NeoVerse (Yang et al., 2026)	8.53	0.29	0.85	8.00	0.35	0.44	9.87	0.33	0.79
StreamSplat (Wu et al., 2026)	15.08	**0.48**	**0.38**	16.30	0.42	0.48	12.37	0.47	0.69
UFO-4D (Hur et al., 2026)	13.76	**0.42**	0.52	16.52	0.32	**0.30**	13.44	0.51	0.43
4DGT (Xu et al., 2025)	8.98	0.21	0.78	9.12	0.43	0.45	8.23	0.21	0.75
FSM-LRM (Ma et al., 2026)	14.35	0.22	0.64	15.50	0.22	0.67	12.45	0.18	0.74
MoVieS (Lin et al., 2026a)	**15.51**	0.37	0.41	**17.13**	**0.44**	**0.28**	**18.29**	**0.51**	**0.35**
Ours w/o CGA	**17.73**	**0.48**	**0.28**	**18.45**	**0.49**	**0.25**	**19.87**	**0.68**	**0.21**
Ours	**17.64**	**0.47**	**0.33**	**17.86**	**0.49**	0.35	**18.94**	**0.62**	**0.31**

vision alone does not explicitly enforce such cross-sample consistency. We therefore use cached predictions as soft targets, analogous to knowledge distillation (Hinton et al., 2015), to regularize overlapping predictions. Thus, cached predictions that preserve information from previous contexts can encourage consistency across overlapping frames. Specifically, for the overlapping frames $\mathcal{O}$ exist in $\mathcal{C}$, we additionally use appearance alignment loss $\mathcal{L}_{\text{aa}}$ and geometry alignment loss $\mathcal{L}_{\text{ga}}$:

$$\mathcal{L}_{\text{aa}} = \sum_{o \in \mathcal{O}} \|I_o - \tilde{I}_o\|_2 + \lambda_{\text{LPS}} \,\text{LPIPS}(I_o, \tilde{I}_o), \mathcal{L}_{\text{ga}} = \sum_{o \in \mathcal{O}} \|\mathcal{D}_o - \tilde{\mathcal{D}}_o\|_1 + \lambda_{\nabla} \|\nabla \mathcal{D}_o - \nabla \tilde{\mathcal{D}}_o\|_1, \quad (3)$$

where $\tilde{I}_o$ and $\tilde{\mathcal{D}}_o$ are cached soft targets, and I_o and $\mathcal{D}_o$ are current predictions. Overall objective is

$$\mathcal{L}_{\text{overall}} = \lambda_d \mathcal{L}_{\text{depth}} + \lambda_m \mathcal{L}_{\text{motion}} + \lambda_r \mathcal{L}_{\text{img}} + \lambda_{\text{aa}} \mathcal{L}_{\text{aa}} + \lambda_{\text{ga}} \mathcal{L}_{\text{ga}}, \quad (4)$$

where all λ balances each objective. During training, we first train FluxSplat sequentially with $\mathcal{L}_{\text{overall}}$ to establish temporally consistent predictions. After some initial sequential training, we then randomly introduce a query $(\mathbf{P}_q, t_q)$ at each timestep and apply Motion-FVS to the cache $\mathcal{C}$ to select $\mathcal{S}_q$, training FluxSplat to predict Gaussians to render queried view for supervision.

Inference. At inference time, each incoming frame triggers two stages for in-distribution streamable DVS, as shown in Fig. 2(left). In the *1st inference* (Fig. 2(a)), at timestep t_τ, we sample $\mathcal{R}_{t_\tau}$ to predict per-frame depth, semantic features, inter-frame motion, and latent states, which are stored in cache $\mathcal{C}$. For an arbitrary query $(\mathbf{P}_q, t_q)$, the *2nd inference* applies Motion-FVS to $\mathcal{C}$ to obtain $\mathcal{S}_q$. Together with their latent states, $\mathcal{S}_q$ is fed into FluxSplat to predict Gaussians at t_q and rendered the queried view from $\mathbf{P}_q$ (Fig. 2(b)). Note that, latent states are kept frozen at the 2nd inference.

5 Experiments

Evaluation Datasets & Metrics. Following recent works (Lin et al., 2026a; Wang et al., 2025c; Yang et al., 2026; Yan et al., 2026; Kim et al., 2026), we evaluate on DyCheck (Gao et al., 2022) (470 timesteps on avg.) and Dynamic NVIDIA (Yoon et al., 2020) (163 timesteps on avg.), two widely used DVS benchmarks. On DyCheck, we use the video sequences captured by a moving camera as input and two static cameras for evaluation. On Dynamic NVIDIA, we follow (Wang et al., 2025c; Wu et al., 2026) and use one camera view as input and the 3 surrounding camera views for evaluation. We also evaluate on Nerfies & HyperNeRF (Park et al., 2021a;b) (241 timesteps on avg.), two widely used DVS datasets, using one camera as input and the other for evaluation, following (Gao et al., 2022). We report PSNR, SSIM, LPIPS from (Gao et al., 2022) for all datasets.

Due to the limited number of scenes in above benchmarks and the potential unreliability of pixel-wise metrics for novel views, we further evaluate generalization on more in-the-wild videos. We construct a benchmark from the entire DAVIS trainval (90 videos) (Pont-Tuset et al., 2017) and 200 random videos from DynPose-100K (Rockwell et al., 2025), covering diverse dynamic scenes and camera trajectories. The benchmark contains 290 videos with 70 timesteps on average. We estimate camera poses using Depth-Anything-3 (Lin et al., 2026b), which also allows us to evaluate FluxSplat under imperfect poses, and randomly fix a pose to synthesize the streams over time. Following (Yang et al., 2026), we evaluate the synthesized videos using selected VBench (Huang et al., 2024b) metrics: Subject Consistency (Cons.), Background Consistency (Cons.), Temporal Flickering (Flick.), Motion Smoothness (Smooth.), Aesthetic Quality (Qual.), and Imaging Quality (Qual.).

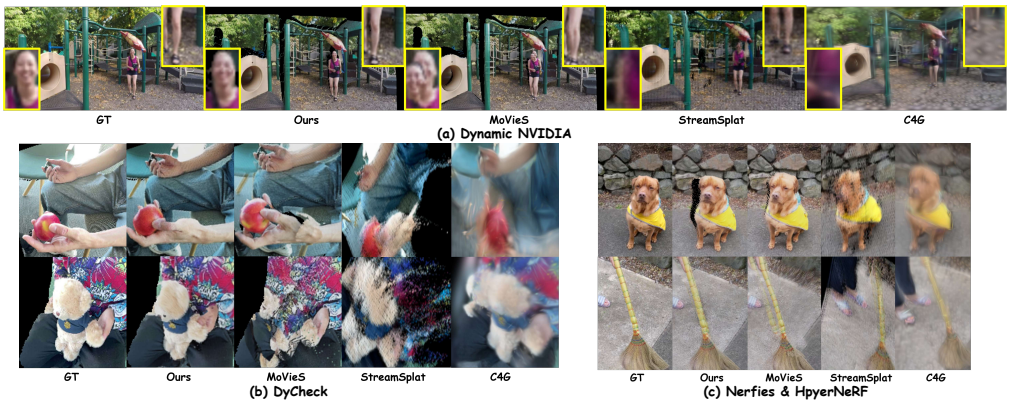

Figure 3: Qualitative results on three benchmarks (Gao et al., 2022; Yoon et al., 2020; Park et al., 2021a;b). We compare with (Lin et al., 2026a; Wu et al., 2026; Kim et al., 2026). (a): yellow boxes highlight same cropped regions with major differences. Invisible regions are rendered as black.

Baselines. We focus on feed-forward DGS methods for streamable DVS and consider two groups of baselines. For the first group, we adapt existing offline feed-forward DGS methods to the streaming setting, including MoVieS (Lin et al., 2026a), 4DGT (Xu et al., 2025), NeoVerse (Yang et al., 2026), C4G (Kim et al., 2026), and FSM-LRM (Ma et al., 2026), without modifying architectures. NeoVerse and C4G employ video diffusion for post-processing; we disable it for a fair comparison. C4G predicts compact Gaussians. When inputs required by Motion-FVS are unavailable, we obtain them using off-the-shelf models (Lin et al., 2026b; Teed & Deng, 2020). For predictions from baselines are not aligned, we apply global alignment. For the second group, we consider methods designed for streamable DVS, including StreamSplat (Wu et al., 2026) and UFO-4D (Hur et al., 2026). These methods only take two adjacent frames as input for the view synthesis. As above baselines do not release their evaluation protocols, we implement the evaluation following descriptions in the papers and publicly available code as closely as possible. Several methods (Liang et al., 2025; Shen et al., 2025; Lin et al., 2025; Yan et al., 2026) are therefore excluded due to unavailable code.

Implementation Details. FluxSplat is built upon (Lin et al., 2026a). We replace its depth prediction head with our HDP head and add our CGA (default $k = 2$) in two Splat heads, while keeping others unchanged. We set $|\mathcal{S}| = |\mathcal{R}| = 12$ for all settings and baselines. We train the model by using AdamW (Loshchilov & Hutter, 2019) with cosine learning rate (lr) scheduling. The lr is set as 4×10^{-5}, model is trained on 8 $\times$ H200 GPUs for approximately five days. We train our model on five datasets: PointOdyssey (Zheng et al., 2023), DynamicReplica (Karaev et al., 2023), Spring (Mehl et al., 2023), VKITTI2 (Cabon et al., 2020), and Stereo4D (Jin et al., 2025). More implementation, model, runtime details are provided in Sections A.1, A.2, A.3.1, respectively.

5.1 Results

Rendering Comparisons at the Original Resolution We compare ours w/o and w/ CGA ($k = 2$) with existing open-source SOTA methods on 3 DVS benchmarks (Yoon et al., 2020; Gao et al., 2022; Park et al., 2021a;b) using resolutions close to those employed during training. Quantitative results in Table 1 show that ours consistently achieve the best overall performance, while qualitative comparisons (Fig. 3) further demonstrate the effectiveness. We compare with C4G (Kim et al., 2026), StreamSplat (Wu et al., 2026), and MoVieS (Lin et al., 2026a), representing compact, online, and offline feed-forward DGS methods. Visualizations are selected from Dynamic NVIDIA, DyCheck, and Nerfies & HyperNeRF, shown in (a), (b), and (c), respectively. Ours produces more accurate and consistent structures for dynamic objects. MoVieS may produce fragmented (Fig. 3(b)) or duplicated structures (Fig. 3(a,c)), while StreamSplat is less robust due to its reliance on adjacent frames, leading to unreliable reconstruction (Fig. 3(b,c)), although it performs reasonably well when novel views are close to the input views (Fig. 3(a)). C4G achieves compactness at the cost of reconstruction quality, producing blurrier results. In contrast, our method provides compact, accurate, and reliable dynamic view rendering.

Table 2: Higher resolution comparison on (Yoon et al., 2020; Gao et al., 2022). Methods receive near-training-resolution images and render at target resolution (Input → Target). DR and US mean direct and upsampled rendering. 1st, 2nd, 3rd best DR results are marked by 1st, 2nd, 3rd.

Method	Dynamic NVIDIA (4×: 238×462 → 952×1848)				DyCheck (2×: 238×238 → 476×476)			
	PSNR ↑	SSIM ↑	LPIPS ↓	Gauss. Num.	PSNR ↑	SSIM ↑	LPIPS ↓	Gauss. Num.
C4G-DR (Kim et al., 2026)	14.25	0.37	0.57	2K	10.45	0.15	0.89	2K
StreamSplat-DR (Wu et al., 2026)	13.30	0.30	0.66	73K	10.89	0.36	0.72	73K
StreamSplat-US	13.33	0.37	0.64	73K	11.23	0.41	0.70	73K
MoVieS-DR (Lin et al., 2026a)	8.96	0.10	0.95	1319K	13.50	0.39	0.72	679K
MoVieS-US	14.54	0.42	0.66	1319K	16.56	0.47	0.54	679K
Ours-DR ($k = 3$)	15.12	0.44	0.53	146K	17.06	0.45	0.39	75K
Ours-DR ($k = 2$)	15.52	0.47	0.46	329K	17.36	0.59	0.37	169K

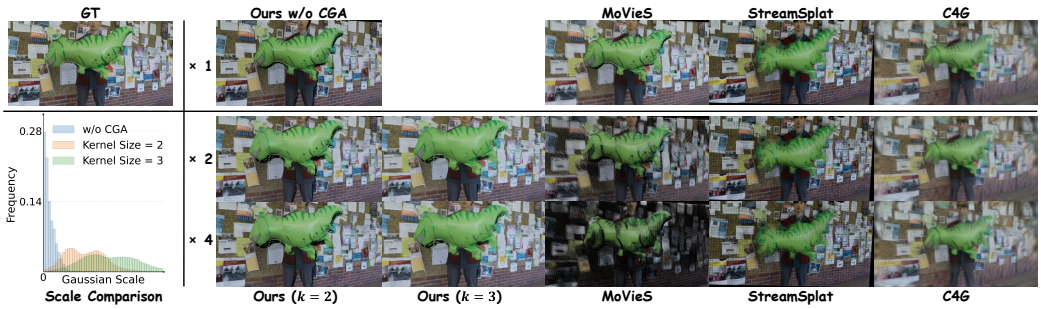

Figure 4: High-resolution qualitative results on dynamic NVIDIA (Yoon et al., 2020). Ours with $k = 2/3$ directly render at $2\times$ and $4\times$ the input resolution with consistent quality. Scale Comparison shows that larger k yield larger scales, while w/o CGA produces scales concentrated near zero.

Rendering Comparisons at the Higher Resolution We evaluate high-resolution (HR) rendering at $2\times$ and $4\times$ the input resolution on DyCheck (Gao et al., 2022) and Dynamic NVIDIA (Yoon et al., 2020). We compare our method ($k = 2/3$) with C4G, StreamSplat, and MoVieS. All of them are trained at low resolutions without HR supervision. Since pixel-wise methods can produce hole artifacts when directly rendering (DR) at HR (see MoVieS in Fig. 4), we additionally report their native-resolution renderings upsampled to the target resolutions for fair comparison. As shown in Table 2, ours consistently achieves the best performance among DR methods, while using only $1/4$ and $1/9$ as many Gaussians as MoVieS at $2\times$ and $4\times$ resolution, respectively. Upsampling improves pixel-wise methods, but our compact representation still achieves better quality. Fig. 4 shows different HR renderings on NVIDIA, whose ground-truth are near-2K. C4G remains blurred, MoVieS suffers increasingly severe hole artifacts due to inaccurate scales, and StreamSplat exhibits slight grid artifacts, whereas ours maintains consistent quality. The scale distribution (Fig. 4 Scale Comparison) further supports this: increasing k in CGA shifts scales toward larger and more accurate values, while w/o CGA concentrates them near zero.

More In-the-Wild Comparisons at the Original Resolution. We additionally compare ours with C4G, StreamSplat, and MoVieS at 210×378 on our in-the-wild benchmark. Quantitative results in Table 3 show that ours consistently outperforms all baselines across all selected VBench metrics, with notable improvements in Background Consistency, Temporal Flickering, and Motion Smoothness, demonstrating more temporally consistent and smoother video synthesis. Qualitative comparisons with MoVieS are shown in Fig. 5. We render novel views at the time of frame I_{TF} using the pose of I_{PF}, and visualize the predicted motion from I_{PF} to I_{TF}. Ours more effectively aggregates Gaussians across timesteps, producing more consistent backgrounds (Fig. 5(up)), while preserving detailed head movements and reconstructing legs more accurately (Fig. 5(bottom)). More qualitative comparisons are provided in Section A.3.3, Figs. 10–11, and our *Supplementary Video*.

Others. See Sections A.3.4 and A.4 for Failure Cases Discussion and Limitations, respectively.

Table 3: Quantitative comparison on our in-the-wild benchmark (Pont-Tuset et al., 2017; Rockwell et al., 2025). The 1st, 2nd, 3rd best results are marked by 1st, 2nd, 3rd.

Method	Subject Cons. ↑	Background Cons. ↑	Temporal Flick. ↑	Motion Smooth. ↑	Aesthetic Qual. ↑	Imaging Qual. ↑
C4G (Kim et al., 2026)	83.1	91.7	96.5	94.7	35.4	23.1
StreamSplat (Wu et al., 2026)	82.6	91.0	95.8	96.7	27.1	44.7
MoVieS (Lin et al., 2026a)	83.4	91.3	96.2	97.0	39.0	50.2
Ours	**84.3**	**92.2**	**97.2**	**97.7**	**39.2**	**50.6**

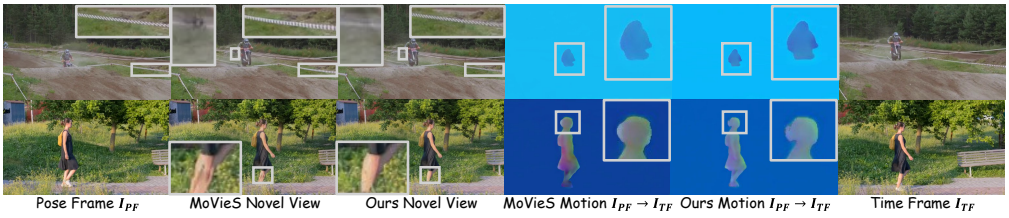

Figure 5: Qualitative comparisons with MoVieS on our In-the-Wild benchmark (Pont-Tuset et al., 2017). Streamable novel views are rendered at the I_{TF}'s timestep using the I_{PF}s' pose and predicted motion from I_{PF} to I_{TF}. Ours makes more consistent background (up), while reconstructing better leg across time and predicting finer-grained head motion (bottom).

Table 4: Ablation studies of temporal (Temp.) and spatial (Spat.) scores, and Relevance Filter (Rele. Fil.) for Motion-FVS sampling.

	Temp.	Spat.	Rele. Fil.	PSNR↑	SSIM↑	LPIPS↓
(1)	✓		✓	18.28	0.55	0.27
(2)		✓	✓	18.39	0.57	0.27
Full	✓	✓	✓	**18.80**	**0.58**	**0.25**
(3)	✓	✓		17.62	0.53	0.28

Table 5: Ablation studies of the three components. HDP Head: History-aware Depth Prediction Head. All: Ours w/o CGA.

	HDP Head	$\mathcal{L}_{aa}$	$\mathcal{L}_{ga}$	PSNR↑	SSIM↑	LPIPS↓
(a)		✓	✓	17.95	0.48	0.26
All	✓	✓	✓	**18.80**	**0.58**	**0.25**
(b)	✓		✓	18.30	0.50	0.26
(c)	✓	✓		17.81	0.48	0.34

5.2 Ablation Studies

We conduct ablations on DyCheck and Nerfies & HyperNeRF, which evaluate dynamic views with static and moving cameras, respectively. We report the average results across the two datasets.

Motion-FVS Ablations. We separately ablate the temporal and spatial diversity of Motion-FVS in rows (1) and (2) of Table 4. Each diversity provides complementary benefits, while jointly considering both of them achieves the best. For row (3), we remove Relevance Filter while keeping both diversity. This leads to inferior performance, indicating that Relevance Filter narrows the search space to query-relevant frames and makes the subsequent sampling more targeted.

Other Ablations. We ablate HDP Head, appearance $\mathcal{L}_{aa}$ and geometry $\mathcal{L}_{ga}$ alignment loss on Ours w/o CGA in rows (a,b,c) of Table 5, respectively. (a) marks importance of keeping a latent state for each timestep to leverage historical information for temporally consistent Gaussian prediction. (b,c) show that two losses improve performance, with $\mathcal{L}_{ga}$ having a larger impact than $\mathcal{L}_{aa}$, suggesting that temporal geometric consistency is more important. See Section A.3.2 for qualitative ablations.

6 Conclusion

We present FluxSplat, a feed-forward dynamic Gaussian Splatting framework for streamable dynamic view synthesis from monocular streams. The History-aware Depth Prediction Head enables effective historical information reuse for better temporally consistent Gaussian prediction, while the Compact Gaussian Adapter improves Gaussian scale prediction, reducing the number of Gaussians and hole artifacts in high-resolution rendering. Our Motion-aware Farthest View Sampling efficiently handles long video streams by selecting informative frames. Experiments show that FluxSplat achieves SOTA performance on three benchmarks and our in-the-wild benchmark.

AI USE STATEMENT

In this work, we used generative AI tools for proofreading, language editing, improving the clarity and readability of the manuscript, and assisting with code debugging. We have not used generative AI tools for generating experimental results or data, developing theoretical models or conceptual frameworks, formulating mathematical claims, designing or providing feedback on research methodology or experiments, implementing the proposed method, or interpreting experimental results. Other required disclosure tasks are not applicable.

We have reviewed and verified all AI-assisted work. We take responsibility for the final content of this work, including all text, claims, results, and artifacts produced with the aid of generative AI.

REFERENCES

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.

Nicolas Ballas, Li Yao, Chris Pal, and Aaron Courville. Delving deeper into convolutional networks for learning video representations. *arXiv preprint arXiv:1511.06432*, 2015.

Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual kitti 2. *arXiv preprint arXiv:2001.10773*, 2020.

Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*, 2016.

Xiaoxue Chen, Ziyi Xiong, Yuantao Chen, Gen Li, Nan Wang, Hongcheng Luo, Long Chen, Haiyang Sun, Bing Wang, Guang Chen, Hongyang Li, Ya-Qin Zhang, Hangjun Ye, and Hao Zhao. Dggt: Feedforward 4d reconstruction of dynamic driving scenes using unposed images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1265–1276, June 2026.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Haiwen Feng, Junyi Zhang, Qianqian Wang, Yufei Ye, Pengcheng Yu, Michael J. Black, Trevor Darrell, and Angjoo Kanazawa. St4rtrack: Simultaneous 4d reconstruction and tracking in the world. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8503–8513, October 2025.

Hang Gao, Ruilong Li, Shubham Tulsiani, Bryan Russell, and Angjoo Kanazawa. Monocular dynamic view synthesis: A reality check. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=pCrB8orUkSq.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

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

Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4220–4230, June 2024a.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. Vbench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21807–21818, June 2024b.

Junhwa Hur, Charles Herrmann, Songyou Peng, Philipp Henzler, Zeyu Ma, Todd Zickler, and Deqing Sun. UFO-4d: Unposed feedforward 4d reconstruction from two images. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=8gDDWqO59H.

Minsik Jeon, Jay Karhade, Deva Ramanan, and Shubham Tulsiani. Point4d: Long-range 4d motion reconstruction. *arXiv preprint arXiv:2609.09145*, 2026.

Linyi Jin, Richard Tucker, Zhengqi Li, David Fouhey, Noah Snavely, and Aleksander Holynski. Stereo4d: Learning how things move in 3d from internet stereo videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10497–10509, June 2025.

Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Dynamicstereo: Consistent dynamic depth from stereo videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13229–13239, June 2023.

Jay Karhade, Nikhil Keetha, Yuchen Zhang, Tanisha Gupta, Akash Sharma, Sebastian Scherer, and Deva Ramanan. Any4d: Unified feed-forward metric 4d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14578–14589, June 2026.

Mungyeom Kim, Minkyeong Jeon, Honggyu An, Jaewoo Jung, Hyuna Ko, Jisang Han, Hyeonseo Yu, Donghwan Shin, Sunghwan Hong, Takuya Narihira, et al. Learning global motion with compact gaussians for feed-forward 4d reconstruction. *arXiv preprint arXiv:2605.31595*, 2026.

Jiahui Lei, Yijia Weng, Adam W. Harley, Leonidas Guibas, and Kostas Daniilidis. Mosca: Dynamic gaussian fusion from casual videos via 4d motion scaffolds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6165–6177, June 2025.

Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *European conference on computer vision*, pp. 71–91. Springer, 2024.

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

Hanxue Liang, Jiawei Ren, Ashkan Mirzaei, Antonio Torralba, Ziwei Liu, Igor Gilitschenski, Sanja Fidler, Cengiz Oztireli, Huan Ling, Zan Gojcic, and Jiahui Huang. Feed-forward bullet-time reconstruction of dynamic scenes from monocular videos. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=oGc1qHAUBJ.

Chenguo Lin, Yuchen Lin, Panwang Pan, Yifan Yu, Tao Hu, Honglei Yan, Katerina Fragkiadaki, and Yadong Mu. Movies: Motion-aware 4d dynamic view synthesis in one second. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 295–306, June 2026a.

Chieh Hubert Lin, Zhaoyang Lv, Songyin Wu, Zhen Xu, Thu Nguyen-Phuoc, Hung-Yu Tseng, Julian Straub, Numair Khan, Lei Xiao, Ming-Hsuan Yang, Yuheng Ren, Richard Newcombe, Zhao Dong, and Zhengqin Li. DGS-LRM: Real-time deformable 3d gaussian reconstruction from monocular videos. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=X2u8esISdb.

Haotong Lin, Sili Chen, Jun Hao Liew, Donny Y. Chen, Zhenyu Li, Yang Zhao, Sida Peng, Hengkai Guo, Xiaowei Zhou, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth anything 3: Recovering the visual space from any views. In *The Fourteenth International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=yirunib8l8.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Ziqiao Ma, Xuweiyi Chen, Shoubin Yu, Sai Bi, Kai Zhang, Chen Ziwen, Sihan Xu, Jianing Yang, Zexiang Xu, Kalyan Sunkavalli, Mohit Bansal, Joyce Chai, and Hao Tan. 4d-LRM: Large space-time reconstruction model from and to any view at any time. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=3i53Z6g6j4.

Ziqiao Ma, Xueyang Yu, Haoyu Zhen, Yuncong Yang, Joyce Chai, and Chuang Gan. Fast spatial memory with elastic test-time training. *arXiv preprint arXiv:2604.07350*, 2026.

Lukas Mehl, Jenny Schmalfuss, Azin Jahedi, Yaroslava Nalivayko, and Andrés Bruhn. Spring: A high-resolution high-detail dataset and benchmark for scene flow, optical flow and stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4981–4991, June 2023.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=a68SUt6zFt. Featured Certification.

Jongmin Park, Minh-Quan Viet Bui, Juan Luis Gonzalez Bello, Jaeho Moon, Jihyong Oh, and Munchurl Kim. Splinegs: Robust motion-adaptive spline for real-time dynamic 3d gaussians from monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26866–26875, June 2025.

Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5865–5874, October 2021a.

Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. Hypernerf: a higher-dimensional representation for topologically varying neural radiance fields. *ACM Trans. Graph.*, 40(6), December 2021b. ISSN 0730-0301. doi: 10.1145/3478513.3480487. URL https://doi.org/10.1145/3478513.3480487.

Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alex Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*, 2017.

Jiawei Ren, Kevin Xie, Ashkan Mirzaei, hanxue liang, Xiaohui Zeng, Karsten Kreis, Ziwei Liu, Antonio Torralba, Sanja Fidler, Seung Wook Kim, and Huan Ling. L4GM: Large 4d gaussian reconstruction model. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=PSPtj26Lbp.

Chris Rockwell, Joseph Tung, Tsung-Yi Lin, Ming-Yu Liu, David F. Fouhey, and Chen-Hsuan Lin. Dynamic camera poses and where to find them. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12444–12455, June 2025.

Johannes L. Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Richard Shaw, Michal Nazarczuk, Jifei Song, Arthur Moreau, Sibi Catley-Chandar, Helisa Dhamo, and Eduardo Pérez-Pellitero. Swings: sliding windows for dynamic 3d gaussian splatting. In *European Conference on Computer Vision*, pp. 37–54. Springer, 2024.

Qiuhong Shen, Xuanyu Yi, Mingbao Lin, Hanwang Zhang, Shuicheng Yan, and Xinchao Wang. Seeing world dynamics in a nutshell. *arXiv preprint arXiv:2502.03465*, 2025.

Colton Stearns, Adam Harley, Mikaela Uy, Florian Dubost, Federico Tombari, Gordon Wetzstein, and Leonidas Guibas. Dynamic gaussian marbles for novel view synthesis of casual monocular videos. In *SIGGRAPH Asia 2024 Conference Papers*, pp. 1–11, 2024.

Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*, pp. 402–419. Springer, 2020.

Narek Tumanyan, Assaf Singer, Shai Bagon, and Tali Dekel. Dino-tracker: Taming dino for self-supervised point tracking in a single video. In *European Conference on Computer Vision*, pp. 367–385. Springer, 2024.

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

Jingkang Wang, Henry Che, Yun Chen, Ze Yang, Lily Goli, Sivabalan Manivasagam, and Raquel Urtasun. Flux4d: Flow-based unsupervised 4d reconstruction. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=FeUGQ6AiKR.

Qianqian Wang, Vickie Ye, Hang Gao, Weijia Zeng, Jake Austin, Zhengqi Li, and Angjoo Kanazawa. Shape of motion: 4d reconstruction from a single video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9660–9672, October 2025c.

Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A. Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10510–10522, June 2025d.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20697–20709, June 2024.

Yiming Wang, Lucy Chai, Xuan Luo, Michael Niemeyer, Manuel Lagunas, Stephen Lombardi, Siyu Tang, and Tiancheng Sun. Learning efficient fuse-and-refine for feed-forward 3d gaussian splatting. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025e. URL https://openreview.net/forum?id=KsmgnL6PUb.

Zike Wu, Qi Yan, Xuanyu Yi, Lele Wang, and Renjie Liao. Streamsplat: Towards online dynamic 3d reconstruction from uncalibrated video streams. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=SaiDRQU7Ez.

Yuxi Xiao, Jianyuan Wang, Nan Xue, Nikita Karaev, Yuri Makarov, Bingyi Kang, Xing Zhu, Hujun Bao, Yujun Shen, and Xiaowei Zhou. Spatialtrackerv2: Advancing 3d point tracking with explicit camera motion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6726–6737, October 2025.

Zhen Xu, Zhengqin Li, Zhao Dong, Xiaowei Zhou, Richard Newcombe, and Zhaoyang Lv. 4DGT: Learning a 4d gaussian transformer using real-world monocular videos. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=qMRFNxioPC.

Zaoming Yan, Qizhou Chen, Yaomin Huang, Pengcheng Lei, Chenhao Shi, Yi Xu, Haichuan Song, and Faming Fang. Taylor-gaussians-flow: Towards non-uniform motion for novel view synthesis from monocular video. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=xGjhMGmmCF.

Jiawei Yang, Jiahui Huang, Boris Ivanovic, Yuxiao Chen, Yan Wang, Boyi Li, Yurong You, Apoorva Sharma, Maximilian Igl, Peter Karkus, Danfei Xu, Yue Wang, and Marco Pavone. STORM: Spatio-temporal reconstruction model for large-scale outdoor scenes. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=M2NFWRPMUd.

Yuxue Yang, Lue Fan, Ziqi Shi, Junran Peng, Feng Wang, and Zhaoxiang Zhang. Neoverse: Enhancing 4d world model with in-the-wild monocular videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 40340–40351, June 2026.

Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20331–20341, June 2024.

Vickie Ye, Ruilong Li, Justin Kerr, Matias Turkulainen, Brent Yi, Zhuoyang Pan, Otto Seiskari, Jianbo Ye, Jeffrey Hu, Matthew Tancik, and Angjoo Kanazawa. gsplat: An open-source library for gaussian splatting. *Journal of Machine Learning Research*, 26(34):1–17, 2025. URL http://jmlr.org/papers/v26/24-1476.html.

Jae Shin Yoon, Kihwan Kim, Orazio Gallo, Hyun Soo Park, and Jan Kautz. Novel view synthesis of dynamic scenes with globally coherent depths from a monocular camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

Shuai Yuan, Yantai Yang, Xiaotian Yang, Xupeng Zhang, Zhonghao Zhao, Lingming Zhang, and Zhipeng Zhang. Infinitevggt: Visual geometry grounded transformer for endless streams. *arXiv preprint arXiv:2601.02281*, 2026.

Chuhan Zhang, Guillaume Le Moing, Skanda Koppula, Ignacio Rocco, Liliane Momeni, Junyu Xie, Shuyang Sun, Rahul Sukthankar, Joëlle K. Barral, Raia Hadsell, Zoubin Ghahramani, Andrew Zisserman, Junlin Zhang, and Mehdi S. M. Sajjadi. Efficiently reconstructing dynamic scenes one d4rt at a time. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7382–7392, June 2026.

Junyi Zhang, Charles Herrmann, Junhwa Hur, Varun Jampani, Trevor Darrell, Forrester Cole, Deqing Sun, and Ming-Hsuan Yang. MonST3r: A simple approach for estimating geometry in the presence of motion. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=lJpqxFgWCM.

Yang Zheng, Adam W. Harley, Bokui Shen, Gordon Wetzstein, and Leonidas J. Guibas. Pointodyssey: A large-scale synthetic dataset for long-term point tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 19855–19865, October 2023.

Kaichen Zhou, Yuhan Wang, Grace Chen, Gaspard Beaudouin, Fangneng Zhan, Paul Pu Liang, and Mengyu Wang. PAGE-4d: VGGT-4d perception via disentangled pose and geometry estimation. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=Nfmzp5PBzr.

Dong Zhuo, Wenzhao Zheng, Jiahe Guo, Yuqi Wu, Jie Zhou, and Jiwen Lu. Streaming visual geometry transformer. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=5APgTKsnx8.

A Appendix

For convenience, the table of contents for the appendix is provided below to facilitate indexing.

Contents

A.1 More Implementation Details 15
A.1.1 Training Datasets 15
A.1.2 Evaluation Datasets & Others 15
A.1.3 Training Details 16
A.2 More Details of Model and Optimization 16
A.2.1 Motion-aware Farthest-View Sampling 16
A.2.2 Optimization 17

A.2.3 Some Detials about Model Architectures 17
A.3 More Results 18
A.3.1 Runtime Analysis 18
A.3.2 Visualization of Ablations 18
A.3.3 More Visualizations 19
A.3.4 Failure Cases Discussion 21
A.4 Limitations and Future Work 21

A.1 More Implementation Details

A.1.1 Training Datasets

We train our model on five datasets, covering both synthetic and real-world scenarios with varying levels of geometric and motion complexity. **(1) PointOdyssey** (Zheng et al., 2023) contains approximately 0.1K synthetic sequences with depth and tracking annotations, and is released under the MIT License. Each sequence is repeated 1000× during training. **(2) DynamicReplica** (Karaev et al., 2023) provides around 0.5K synthetic sequences with depth and tracking annotations under the Attribution-NonCommercial 4.0 International License, with each sequence repeated 100×. **(3) Spring** (Mehl et al., 2023) consists of approximately 0.03K synthetic sequences with depth annotations but no explicit tracking annotations, and is released under the CC BY 4.0 license. Each sequence is repeated 2000× during training. **(4) VKITTI2** (Cabon et al., 2020) contains around 0.1K synthetic sequences with depth annotations and is distributed under the Creative Commons Attribution-NonCommercial-ShareAlike 3.0 License, with each sequence repeated 500×. Finally, **(5) Stereo4D** (Jin et al., 2025) is a large-scale real-world dataset with approximately 98K sequences and 19.6M frames, providing both depth and tracking annotations under the CC0 1.0 Universal license. Each sequence is used once without repetition. Together, these datasets provide diverse coverage of scene appearance, object motion, camera motion, and geometric structure for training our feed-forward DGS model.

A.1.2 Evaluation Datasets & Others

DyCheck (Gao et al., 2022). As described in Shape-of-Motion (Wang et al., 2025c), the camera poses provided by the original dataset are inaccurate. We therefore use the refined camera poses released by Shape-of-Motion. We evaluate on five scenes: `Apple` (475 timesteps), `Block` (350 timesteps), `Paper-Windmill` (277 timesteps), `Spin` (426 timesteps), and `Teddy` (350 timesteps). All videos are provided at 960×480 resolution.

Dynamic NVIDIA (Yoon et al., 2020). We follow Shape-of-Motion (Wang et al., 2025c) for preprocessing and camera selection. We evaluate on seven scenes: `Skating` (191 timesteps), `Jumping` (191 timesteps), `Balloon1` (95 timesteps), `Balloon2` (143 timesteps), `Playground` (141 timesteps), `Truck` (203 timesteps), and `Umbrella` (179 timesteps). Specifically, we use footage from static camera 4 as input and cameras 3, 5, and 6 for novel-view evaluation. The videos have a maximum resolution of 1920×1080.

Nerfies & HyperNeRF. Since Nerfies (Park et al., 2021a) and HyperNeRF (Park et al., 2021b) were captured using the same camera rig, we combine the two datasets under a unified evaluation protocol. From Nerfies, we use `Broom` (197 timesteps), `Curls` (57 timesteps), `Tail` (238 timesteps), and `Toby-Sit` (308 timesteps), while from HyperNeRF, we select `3DPrinter` (207 timesteps), `Chicken` (164 timesteps), and `Peel-Banana` (513 timesteps). Following the setup provided in DyCheck (Gao et al., 2022), we use one camera as input and another camera for novel-view evaluation. All videos are provided at $480p$ resolution.

Other Details. During evaluation, if any co-visible masks are provided, we use them for rendering and evaluation. As a result, our PSNR, SSIM, and LPIPS in this paper represent mPSNR, mSSIM, and mLPIPS, which follows the previous work (Gao et al., 2022). For rendering comparisons at the orignial resolution, the input and rendered resolutions for DyCheck, Dynamic NVIDIA, and Nerfies & HyperNeRF are 238×238, 238× 462, and 266×476, respectively. For DyCheck and

Nerfies & HyperNeRF, we follow MoVieS and NeoVerse and crop a portion of the area from the original image using the above resolution. For high-resolution evaluation, we only evaluate on DyCheck and Dynamic NVIDIA. For DyCheck, we crop the corresponding region from the high-resolution images according to the crop used in the original-resolution setting, while preserving the same area for a fair comparison. For Dynamic NVIDIA, we directly use the undistorted high-resolution images. For offline feed-forward DGS models (Lin et al., 2026a; Yang et al., 2026; Kim et al., 2026; Ma et al., 2026; Xu et al., 2025), new frames are progressively fed into the models in an auto-regressive manner during streaming evaluation. This progressive accumulation increases the computational and memory footprint and inference latency as the stream grows. As the accumulated context reaches the practical limit, we use Motion-FVS to selectively maintain informative frames within a fixed budget, allowing the stream to be extended substantially while keeping inference memory consumption bounded.

A.1.3 Training Details

Training Procedure. During training, we train FluxSplat for 13 epochs in total. For the first 3 epochs, we train FluxSplat sequentially with $\mathcal{L}_{\text{overall}}$ to establish temporally consistent predictions, without introducing random queries. For the subsequent epochs, we randomly introduce a query $(\mathbf{P}_{\text{q}}, t_{\text{q}})$ at each timestep and apply Motion-FVS to select $\mathcal{S}_q$ from the cache, training FluxSplat to predict Gaussians for rendering the queried view under the corresponding supervision. We use a temporal window of 16 frames, while the temporal offset between the query pose frame and the query time frame is randomly sampled, typically spanning more than 2 frames and up to the current stream length. After 10 epochs, we freeze the backbone and HDP head, including the depth and motion predictions provided to CGA, and add CGA to the canonical and deformable splat heads. We then train the resulting model for 3 additional epochs, during which only the newly introduced CGA modules are optimized.

Training Settings. All training images are resized to 224×224 pixels. To improve memory and computational efficiency, we employ the `gsplat` (Ye et al., 2025) rendering backend, gradient checkpointing (Chen et al., 2016), gradient accumulation, and `bfloat16` mixed-precision training. For the multi-task objective, we set the loss weights based on the relative scales of the individual loss terms rather than performing extensive manual tuning. Specifically, we use $\lambda_d = \lambda_r = \lambda_{pt} = 1$, $\lambda_m = \lambda_{dist} = 10$, and $\lambda_{aa} = \lambda_{ga} = 0.5$, which yields comparable magnitudes across the different objectives. We set $\lambda_{\text{ts}} = 2.0$ as the default for Motion-FVS. The input timesteps of all sample frames are normalized by min-max normalization based on their frame index. We set $|\mathcal{R}| = |\mathcal{S}| = 12$, the length of temporal window on sampling for $\mathcal{R}$ is 16. We set CGA with $k = 2$ as the default setting. We use the same data preprocessing and camera conventions as the MoVieS.

A.2 More Details of Model and Optimization

We will release the code of our model after acceptance. Below, we describe more detials.

A.2.1 Motion-aware Farthest-View Sampling

Relevance Filtering. Following DINO-Tracker (Tumanyan et al., 2024), for the cosine similarity score of semantic features between $I_{t_{\mathbf{P}}}$ and each temporally searched frame I'_t (in interval from $I_{t_{\mathbf{P}}}$ to I_{t_q}) as:

$$\eta_{\text{sem}}(I_{t_{\mathbf{P}}}, I'_t) = \frac{\mathbf{f}_{t_{\mathbf{P}}}^{\top} \mathbf{f}'_t}{\|\mathbf{f}_{t_{\mathbf{P}}}\|_2 \|\mathbf{f}'_t\|_2}, \qquad \mathbf{f}_{t_{\mathbf{P}}} = \phi_{\text{dino}}(I_{t_{\mathbf{P}}}), \qquad \mathbf{f}'_t = \phi_{\text{dino}}(I'_t). \tag{5}$$

where $\phi_{dino}(\cdot)$ denotes the DINOv2 (Oquab et al., 2024) feature extractor. Note that, following DINO-Tracker (Tumanyan et al., 2024), we use the feature of the 16th layer in the DINOv2 and set $\eta_{\text{sem}} = 0.7$ for convenience.

Motion-Guided Sampling. For the mean motion magnitude v of any adjacent frames (I_t, I_{t+1}), we follow previous work (Shaw et al., 2024) and compute

$$v = \texttt{M3}(I_t, I_{t+1}) = \frac{1}{HW} \sum_{z=1}^{HW} \left\| f_{\text{motion}}\left(I_t^{(z)}, I_{t+1}^{(z)}\right) \right\|_2, \tag{6}$$

where, $\texttt{M3}(\cdot,\cdot)$ means the mean motion magnitude function of adjacent frames, $f_{\text{motion}}(\cdot,\cdot)$ denotes the motion estimation function, and HW is the total number of points. When computing each temporal interval in $\mathcal{S}_{\text{tmp}}$, for non-consecutive frames, accumulated motion is obtained from the recorded adjacent-frame motion, with linear interpolation used when intermediate observations are unavailable. Note that, for the final score (line 8 in Alg. 1), we independently normalize the spatial and temporal diversity scores over the current candidate set before combining them, with λ_{ts} controlling their relative contributions.

A.2.2 Optimization

Our model is trained with the losses used in prior work (Lin et al., 2026a), including depth loss $\mathcal{L}_{\text{depth}}$, motion loss $\mathcal{L}_{\text{motion}}$, and rendering loss $\mathcal{L}_{\text{img}}$. To further enforce consistency across timesteps, we introduce an Appearance Alignment Loss $\mathcal{L}_{\text{aa}}$ and a Geometry Alignment Loss $\mathcal{L}_{\text{ga}}$. For $\mathcal{L}_{\text{aa}}$ and $\mathcal{L}_{\text{ga}}$, please check in the main paper. Below we describe the depth loss $\mathcal{L}_{\text{depth}}$, motion loss $\mathcal{L}_{\text{motion}}$, and rendering loss $\mathcal{L}_{\text{img}}$.

The depth loss supervises both the predicted depth maps and their spatial gradients. Specifically, we compute the mean squared error between the rendered depth $\mathcal{D}_e$ and the ground-truth depth $\hat{\mathcal{D}}_e$, and add an ℓ_1 penalty on the gradient difference, after masking out invalid depth regions:

$$\mathcal{L}_{\text{depth}} = \sum_{e=1}^{E} (\|\mathcal{D}_e - \hat{\mathcal{D}}_e\|_2 + \|\nabla\mathcal{D}_e - \nabla\hat{\mathcal{D}}_e\|_1). \tag{7}$$

The rendering loss encourages photometric fidelity between the 3D Gaussian splatting outputs and the input video frames. It combines an ℓ_2 reconstruction term with a perceptual LPIPS loss over the $|E|$ sampled camera views:

$$\mathcal{L}_{\text{img}} = \sum_{e=1}^{E} (\|I_e - \hat{I}_e\|_2 + \lambda_{\text{LPS}}\,\text{LPIPS}(I_e, \hat{I}_e)), \tag{8}$$

where, $\lambda_{\text{LPS}} = 0.5$, and I_e and $\hat{I}_e$ denote the rendered and ground-truth image, and $E \in \{|\mathcal{R}|, |\mathcal{S}|\}$.

The motion loss regularizes 3D point trajectories derived from the dynamic scene representation. Given tracked points visible across frames, we define ground-truth displacements Δx_i between two timestamps. The loss comprises a pointwise ℓ_1 alignment term and a distribution-level term that preserves the pairwise inner-product structure of the motion field, promoting sparsity and relative distance consistency:

$$\mathcal{L}_{\text{motion}} = \lambda_{\text{pt}}\mathcal{L}_{\text{pt}} + \lambda_{\text{dist}}\mathcal{L}_{\text{dist}}, \tag{9}$$

where,

$$\mathcal{L}_{\text{pt}} = \frac{1}{P}\sum_{i\in\Omega} \|\Delta\hat{x}_i - \Delta x_i\|_1, \tag{10}$$

$$\mathcal{L}_{\text{dist}} = \frac{1}{P^2}\sum_{(i,j)\in\Omega\times\Omega} \|\Delta\hat{x}_i \cdot \Delta\hat{x}_j^\top - \Delta x_i \cdot \Delta x_j^\top\|_1, \tag{11}$$

and Ω is the set of P valid tracked points. The weighting factors are set to $\lambda_{\text{pt}} = 1$, $\lambda_{\text{dist}} = 10$, and the overall motion weight $\lambda_m = 10$ in the combined objective. All loss terms are summed with unit weight for depth and rendering, except for the LPIPS component as specified above.

A.2.3 Some Detials about Model Architectures

For the GRU architecture in our model, we follow the design of previous work (Ballas et al., 2015), adopting a convolutional recurrent structure to maintain spatially-aware temporal features. We also initialize an latent state for each timestep. Our Compact Gaussian Adapter consists of three convolutional layers and is designed to efficiently transform per-pixel features into a compact representation. For the remaining model architectures, we followed MoVieS.

Table 6: Inference speed and GPU memory of our w/o and w CGA, when the number of sampled frames is 12.

Method	Inference Speed	GPU Memory
Ours w/o CGA	616.49 ms	9.09 GB
Ours	864.40 ms	10.62 GB

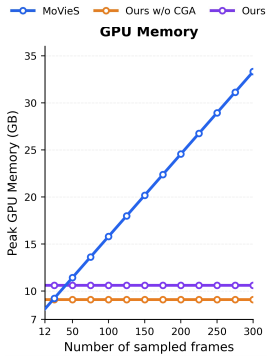

Figure 6: GPU memory scaling with the number of sampled frames. We evaluate Ours, Ours w/o CGA, and MoVieS (Lin et al., 2026a) with an increasing number of sampled frames from 12 to 300. Without using our Motion-FVS, the GPU memory of MoVieS increases approximately linearly with the number of sampled frames, while Ours w/o CGA and Ours maintain fixed and low memory costs.

A.3 More Results

A.3.1 Runtime Analysis

To further investigate the scalability with respect to video length, we evaluate streamable DVS on the video stream ranging from 12 to 300 frames (resolution: 210×378). For our methods, we use a fixed budget of 12 sampled frames for Motion-FVS. We report the average inference speed and GPU inference memory consumption of Ours and Ours w/o CGA in Table 6. Adding CGA introduces additional computational and memory overhead due to its local feature interaction operations. Nevertheless, this overhead improves Gaussian spatial coverage and mitigates hole artifacts, particularly in high-resolution rendering. Despite this overhead, our method maintains nearly constant inference memory consumption at around 10 GB as the video length increases, enabling streamable DVS even on GPUs with low GPU memory (such as, 11 GB). Moreover, the number of sampled frames can be adjusted according to the available GPU memory, allowing our method to flexibly operate across devices with different memory capacities.

We further analyze how GPU memory consumption scale with video length in Fig. 6. Compared with MoVieS w/o Motion-FVS, whose GPU memory consumption increase approximately linearly with video length as more frames are processed, our streamable design maintains a fixed 12-frame input through Motion-FVS, resulting in nearly constant inference GPU memory consumption as the video length increases. Moreover, our inference time remains substantially more stable than that of MoVieS as the stream grows. For example, when the number of streaming frames reaches 80, MoVieS requires 2,096.07 ms for inference, whereas our method remains at around 800 ms. This bounded input size enables our method to process long video streams without increasing resource consumption.

A.3.2 Visualization of Ablations

To further characterize the role three components (HDP Head, two losses) in our framework, we provide qualitative comparisons on in-the-wild videos in Fig. 7. Without the HDP Head (Fig. 7(a)), the reconstructed dynamic region (the tennis player's feet) becomes incomplete, while the background also becomes inconsistent across different pose and time frames. For the loss ablations, removing $\mathcal{L}_{aa}$ ((Fig. 7(b))) leads to darker tones in both the dynamic region and background, whereas remov-

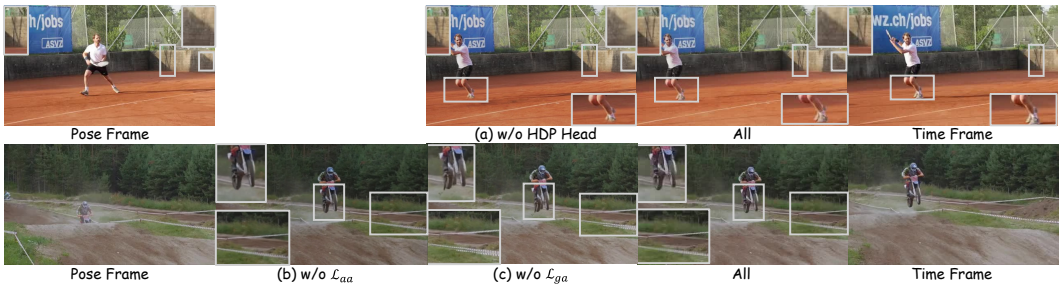

Figure 7: Qualitative comparisons of ablated structures and losses on in-the-wild videos (Pont-Tuset et al., 2017). The image labels (a-c) correspond to the row numbers (a-c) in Table 5. Pose Frame and Time Frame denote the pose used for rendering and the scene dynamic being rendered, respectively. (a) w/o HDP Head shows the incomplete feet. It also shows inconsistent background compared with same region in Pose Frame and Time Frame. Using all components reconstructs the leg completely and shows consistent background. In (b), w/o $\mathcal{L}_{aa}$, darker tones appear in the motorcycle and background, while (c) w/o $\mathcal{L}_{ga}$ removes these tonal discrepancies but introduces ghosting artifacts. Using all components achieves both better reconstruction of dynamic regions and more faithful tones.

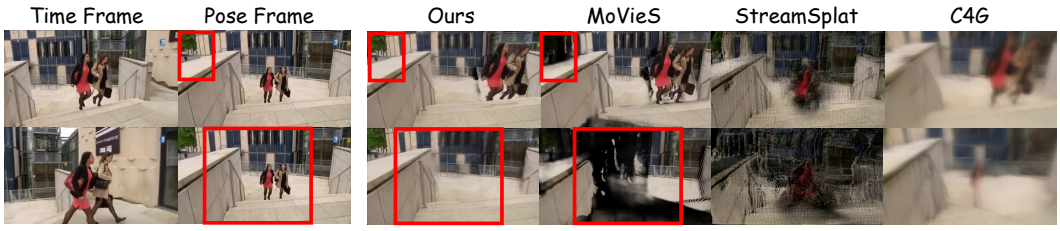

Figure 8: Qualitative comparisons on our in-the-wild benchmark (Pont-Tuset et al., 2017; Rockwell et al., 2025). We fix the camera pose and rendering at two different timesteps. We can clearly see that our results show more consistent and better background and dynamic parts than others.

ing $\mathcal{L}_{ga}$ (Fig. 7(c)) alleviates these tonal discrepancies but introduces ghosting artifacts. The Full model avoids all above issues, achieving more complete dynamic reconstruction with more faithful appearance. The temporal consistency of novel views rendered from the predicted Gaussians can be further observed in Fig. 8, where our model produces consistent background renderings at different timesteps from the same camera pose. This demonstrates that the predicted Gaussians maintain consistent scene appearance and geometry over time.

A.3.3 More Visualizations

We provide additional qualitative video comparisons in the "`Supplementary Video Material`", where we compare with MoVieS, StreamSplat, and C4G on streamable DVS. We further provide extensive visualizations on our in-the-wild benchmark (Pont-Tuset et al., 2017; Rockwell et al., 2025), covering both dynamic and static scenes (only serval), as well as different camera poses and timesteps. ***Note that***, for all the visualizations below, we use the camera pose of the Pose Frame to render the scene at the time of Time Frame.

Two Timesteps. We visualize the novel view at two different timesteps while fixing the camera pose, as shown in Fig. 8. Our method produces more temporally consistent reconstructions, with improved fidelity in both the background and dynamic regions compared with the other methods.

Static Scenes. We further visualize our method on two static scenes from the in-the-wild benchmark, as shown in Fig. 9. We continuously feed the video stream of a static scene into the model

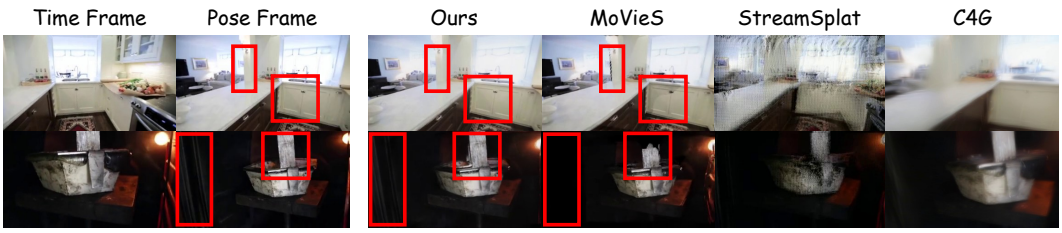

Figure 9: Qualitative comparisons of static scenes on our in-the-wild benchmark (Pont-Tuset et al., 2017; Rockwell et al., 2025). Red boxes highlight major differences. We input video streams of static scenes into our method, and our method still reconstructs the scene well and maintains good image quality, even in low-light environments.

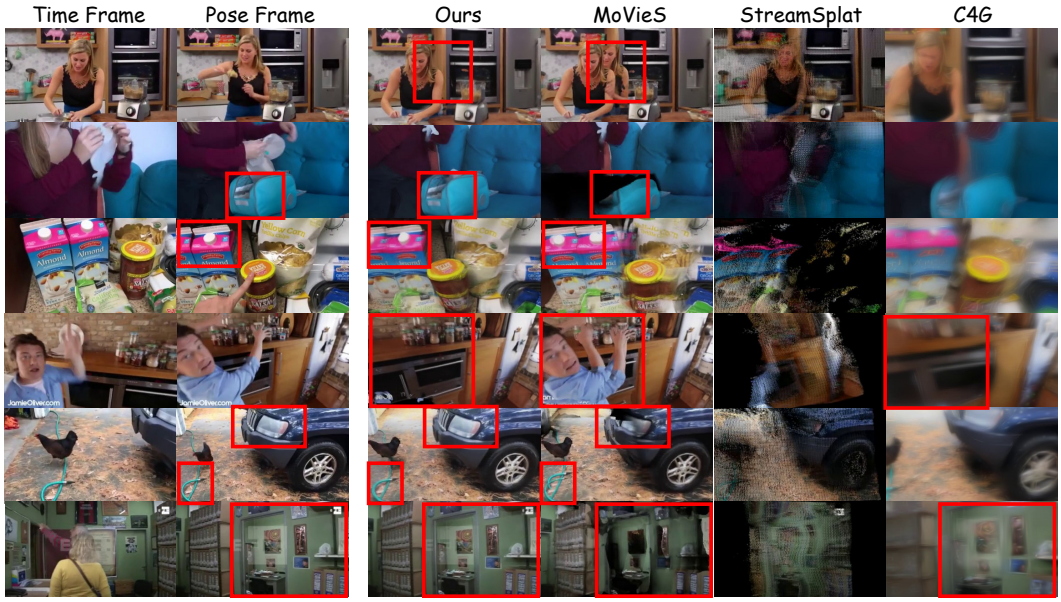

Figure 10: Qualitative comparisons on our in-the-wild benchmark (Pont-Tuset et al., 2017; Rockwell et al., 2025). We render image at the time of the Time Frame from the camera pose of the Pose Frame. Red boxes highlight major differences. Our method consistently delivers better image quality from novel view across a variety of object dynamics, camera motion, and scene conditions.

as if it were a dynamic scene, while observing the scene from a fixed camera pose. Despite the absence of scene motion, our method reconstructs the static scene well and maintains high image quality, including under challenging low-light conditions. The red boxes highlight representative differences between our method and the competing approaches.

Additional In-the-Wild Comparisons. Figures 10 and 11 present additional qualitative comparisons across a diverse set of in-the-wild scenes, including different kinds of object dynamics, camera motion, and scene conditions. Our method consistently achieves better novel-view image quality across various conditions. The red boxes highlight representative regions where our method produces clearer details and fewer reconstruction artifacts (e.g., ghosting, holes, overlapping, distortion, blurring, and grid artifacts) than competing methods.



Figure 11: Qualitative comparisons on our in-the-wild benchmark (Pont-Tuset et al., 2017; Rockwell et al., 2025). We render image at the time of the Time Frame from the camera pose of the Pose Frame. Red boxes highlight major differences. Our method consistently delivers better image quality from novel view across a variety of object dynamics, camera motion, and scene conditions.



Figure 12: Visualization of failure cases on our in-the-wild benchmark (Pont-Tuset et al., 2017; Rockwell et al., 2025). Our method may struggle in challenging scenes with multiple moving objects or limited illumination, resulting in degraded novel-view rendering quality and reconstruction.

A.3.4 Failure Cases Discussion

While our method demonstrates robust performance across a wide range of dynamic scenes, it still encounters challenging cases. As shown in Fig. 12, ours may struggle in scenes containing a large number of moving objects. Degraded performance under limited illumination may occur, where limited visual information makes accurate motion estimation and dynamic scene reconstruction more difficult. These failure cases highlight the limitations of our current approach. Notably, such challenging conditions also remain difficult for all existing methods, suggesting that they represent common limitations of current approaches rather than issues specific to our method.

A.4 Limitations and Future Work

Our Motion-FVS currently assumes known camera poses. Although we use Depth-Anything-3 (Lin et al., 2026b) to estimate camera poses for evaluation, the pose estimation itself is not integrated into our streaming pipeline. Integrating online pose estimation could simplify the overall pipeline to improve robustness. The frames selected by Motion-FVS also depends on sampled-frame coverage, as large inter-frame motion or insufficient FoV overlap can lead to unreliable reconstruction. More

effective sampling or compression strategies, potentially learning-based, could further improve the robustness of long-sequence processing.

Efficiency remains a limitation of our framework. While our design bounds inference memory with a fixed frame budget, the current inference latency is still too high to support real-time streaming. Reducing the inference latency while maintaining the bounded-memory property remains an important direction for future work.

Finally, while our model supports high-resolution rendering, processing high-resolution inputs remains memory-intensive. More efficient input-processing strategies are therefore needed to improve the practicality and scalability for real-time high-resolution applications.