# LIFESTREAMGS: LIFECYCLE-AWARE STREAMING 3D GAUSSIAN SPLATTING FOR UNPOSED VIDEOS

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

## ABSTRACT

Streaming feed-forward 3D Gaussian Splatting (3DGS) reconstructs renderable scenes from unposed video streams. In this setup, each frame's Gaussians are predicted without future observations and remain fixed as new frames arrive, causing prediction errors to accumulate. Dense per-pixel predictions further amplify this problem by adding redundant Gaussians, which increasingly degrade rendering quality over long sequences. To this end, we propose LifeStreamGS, a lifecycle-aware framework that treats dense Gaussian predictions as candidates and controls their placement, retention, and contribution to rendering. These decisions are implemented through merge-guided Gaussian proposal, existence-aware candidate validation, and view- and time-aware opacity aging, reducing redundancy and the impact of accumulated errors. Furthermore, for long videos, we extend this framework through chunk-wise streaming, reconstructing overlapping windows and aligning their reconstructions into a shared coordinate system. Experiments show competitive novel-view synthesis quality with lower runtime and GPU memory usage than the evaluated optimization-based baselines in most tested settings, while supporting 500-view reconstruction and online video stabilization.

## 1 INTRODUCTION

3D Gaussian Splatting (3DGS) enables high-quality, real-time novel view synthesis (NVS) Kerbl et al. (2023), but typically requires iterative optimization for each scene Yu et al. (2024); Huang et al. (2024). Recent feed-forward methods Charatan et al. (2024); Ye et al. (2024); Jiang et al. (2025); Liu et al. (2025) reduce this cost by directly predicting Gaussians from input images, with pose-free methods further removing the need for known camera poses. Most of these methods, however, assume all input views are available before reconstruction, making them unsuitable for streaming applications where frames arrive one at a time, and the renderable scene must be updated as new observations become available. This motivates streaming feed-forward 3DGS from long unposed videos, where reconstruction proceeds without future frames or per-scene optimization.

This setting introduces two limitations. First, Gaussians predicted for the current frame cannot benefit from future views, which may reveal occluded regions or provide additional constraints on geometry and appearance. Second, in the feed-forward accumulation setting considered here, previously generated Gaussians remain fixed rather than being refined using later observations. Consequently, even when later frames revisit the same scene regions, errors in previously predicted Gaussians remain uncorrected and can continue to affect rendering as new Gaussians are added.

As illustrated in Fig. 1, dense per-pixel Gaussian prediction leads to overlapping primitives that can amplify the impact of accumulated prediction errors on rendering. Successive frames often observe the same scene regions, yet each frame produces another dense set of Gaussians. This repeatedly represents the same surfaces with primitives whose geometry and appearance may differ. Retaining all of them therefore increases representation size and can produce ghosting artifacts during rendering. Long sequences further amplify this problem by accumulating more overlapping predictions. Effective streaming reconstruction thus requires control over which predictions enter the scene and how they contribute to later views.

Our key observation is that dense per-pixel predictions should serve as candidates, rather than all being retained and contributing equally to rendering. We introduce LifeStreamGS, a lifecycle-aware framework built around three successive decisions. Merge-guided Gaussian proposal determines



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

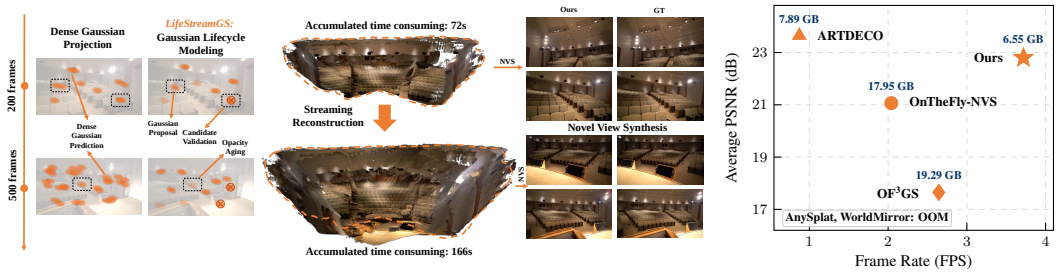


Figure 1: **Left:** Orange ellipses schematically illustrate Gaussian projections onto the image plane. Gaussian lifecycle modeling controls proposal, validation, and rendering contribution to reduce redundant accumulation. LifeStreamGS supports streaming reconstruction and NVS as the input grows from 200 to 500 frames, with cumulative runtimes shown. **Right:** Speed–quality–memory trade-off at 128 input views; labels above the markers report peak GPU memory in GB.

where to place Gaussians by selecting candidate locations from dense predictions within local image patches. Existence-aware candidate validation then decides which candidates to retain, controlling the Gaussians added to the scene. However, a retained Gaussian may still be less reliable for some rendering views than others. View- and time-aware opacity aging therefore adjusts its rendering contribution according to the viewing direction and temporal distance from its source observation. Together, these stages control Gaussian placement, retention, and rendering contribution, reducing redundant accumulation and limiting the impact of accumulated prediction errors. For long videos, we reconstruct overlapping chunks and align their reconstructions into a common coordinate system. With Gaussian lifecycle modeling applied within each chunk, LifeStreamGS reduces redundant accumulation and limits its impact on rendering over long sequences. Our contributions are threefold:

- We introduce LifeStreamGS, a streaming feed-forward framework for reconstructing renderable 3D Gaussian scenes from long unposed videos, addressing the accumulation of fixed, potentially inaccurate per-frame predictions.
- We propose Gaussian lifecycle modeling through merge-guided Gaussian proposal, existence-aware candidate validation, and view- and time-aware opacity aging, controlling where Gaussians are placed, which are retained, and how they contribute to rendering.
- We develop chunk-wise streaming to extend Gaussian lifecycle modeling to long videos, reconstructing overlapping windows and aligning their reconstructions through shared frames into a globally renderable scene.

Experiments demonstrate competitive NVS quality with favorable runtime and GPU memory usage, including reconstruction from 500-view streams and an application to online video stabilization.

## 2 Related Work

**Feed-forward-based offline 3DGS.** Generalizable 3DGS predicts Gaussian primitives without per-scene optimization. Calibrated methods such as PixelSplat Charatan et al. (2024) and MVSplat Chen et al. (2024) use known cameras; pose-free methods including NoPoSplat Ye et al. (2024), FLARE Zhang et al. (2025), and AnySplat Jiang et al. (2025) jointly infer cameras and geometry; and recent extensions improve view alignment, target-view reasoning, or long-sequence representation Bui et al. (2026); Kim et al. (2026); Itkin et al. (2026); Ren et al. (2026). These offline methods usually enable later views to influence earlier predictions, whereas online 3DGS must update from past observations only.

**Optimization-based online 3DGS.** Optimization-based systems such as OnTheFly-NVS Meuleman et al. (2025), ARTDECO Li et al. (2025a), Gaussian On-the-Fly Splatting Xu et al. (2025), and LongSplat Lin et al. (2025a) progressively insert frames and refine poses, geometry, and Gaussians, sometimes using camera or matching priors. Later frames can therefore correct earlier errors, but iterative refinement and global alignment increase latency and complexity. We instead target feed-forward online reconstruction.

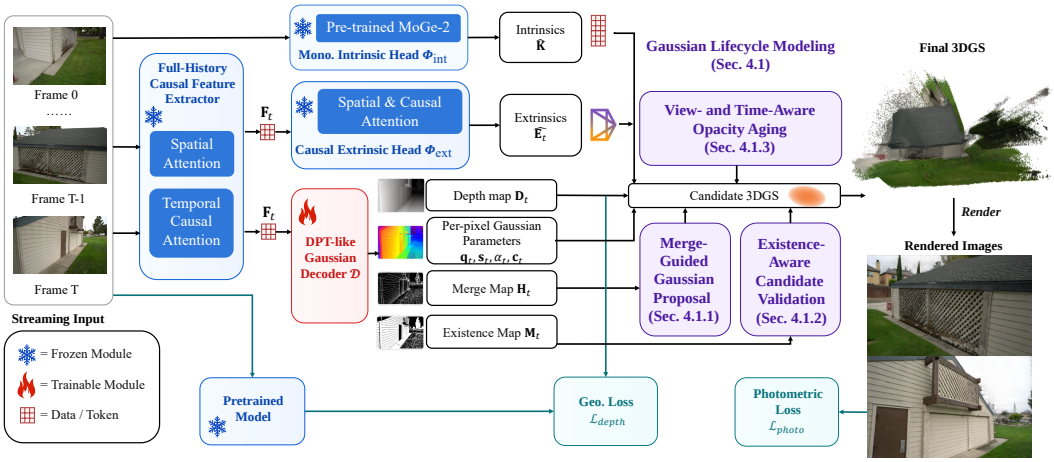


Figure 2: Overview of LifeStreamGS. The framework processes unposed video streams causally, recovers camera poses, and manages dense per-pixel Gaussian candidates through merge-guided proposal, existence-aware validation, and opacity aging.

**Feed-forward-based online 3DGS.** StreamGS Li et al. (2025b), SaLon3R Guo et al. (2025), and OF$^3$GS Chen et al. (2026) process streams or long inputs without per-scene optimization. StreamSplat Wu et al. (2025) requires the later endpoint frame for temporal interpolation, while ReCoSplat Cheng et al. (2026) jointly processes frames within each chunk, allowing earlier-frame predictions to use later frames in the same chunk. In contrast, LifeStreamGS preserves frame-wise causal prediction while combining lifecycle modeling and chunk-wise streaming for reliable, scalable online NVS.

## 3 Problem Formulation

Given an unposed video stream $\mathcal{I} = \{\mathbf{I}_t\}_{t=0}^{T-1}$ of $T$ frames, online 3DGS maintains an accumulated renderable Gaussian set $\mathcal{G}_{\leq t}$ after processing frames $\mathcal{I}_{0:t}$. Each Gaussian $\mathbf{g}_i = \{\boldsymbol{\mu}_i, \mathbf{q}_i, \mathbf{s}_i, \alpha_i, \mathbf{c}_i\} \in \mathcal{G}_{\leq t}$ contains a center, rotation, scale, opacity, and color, respectively. The camera parameters $\mathcal{P}_t = \{\mathbf{K}, \mathbf{R}_t, \mathbf{t}_t\}$ comprise intrinsics $\mathbf{K}$, fixed within the stream, and frame-dependent extrinsic rotation $\mathbf{R}_t$ and translation $\mathbf{t}_t$.

At time $t$, with no future frames available, the current frame $\mathbf{I}_t$ and the feature-context state $\mathcal{S}_{t-1}$ from preceding frames are the inputs to one causal forward pass, whose current-frame Gaussian output $\mathcal{G}_t$ is then appended to the accumulated set:

$$\begin{aligned}(\mathcal{G}_t, \mathcal{P}_t, \mathcal{S}_t) &= f_\theta(\mathbf{I}_t, \mathcal{S}_{t-1}),\\ \mathcal{G}_{\leq t} &= \mathcal{G}_{\leq t-1} \cup \mathcal{G}_t.\end{aligned} \tag{1}$$

Here, $f_\theta$ is the frame-wise prediction function with learnable parameters $\theta$, and $\mathcal{S}_t$ is the updated feature-context state carried to the next frame.

## 4 Proposed Method

Since previously predicted Gaussians are not refined using later observations, LifeStreamGS controls the admission and rendering contributions of new predictions to limit the impact of accumulated errors. The framework uses merge-guided Gaussian proposal to determine candidate locations from dense predictions, existence-aware candidate validation to decide which candidates are added to $\mathcal{G}_t$, and view- and time-aware opacity aging to adjust the accumulated Gaussians' rendering contributions (Sec. 4.1). For long videos, chunk-wise streaming applies Gaussian lifecycle modeling within overlapping windows and aligns their reconstructions into a global Gaussian set (Sec. 4.2).

Specifically, as shown in Fig. 2, following StreamVGGT Zhuo et al. (2025), we use a causal feature extractor to combine the current image $\mathbf{I}_t$ with historical tokens in the feature-context state $\mathcal{S}_{t-1}$ and

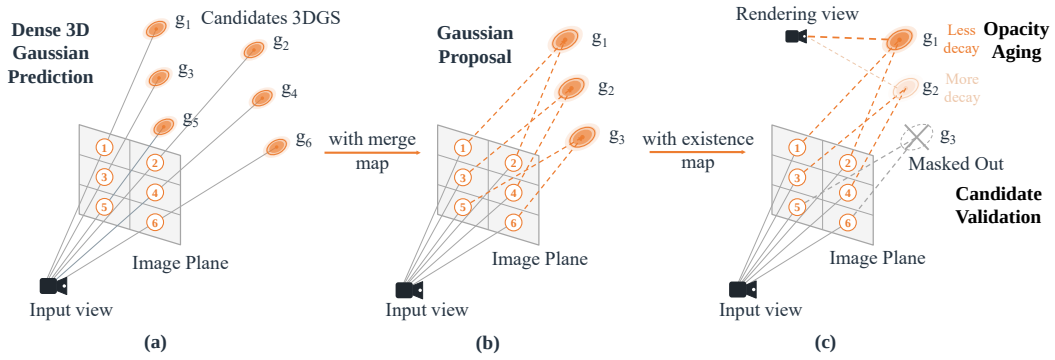


Figure 3: Schematic of Gaussian lifecycle modeling. Dense pixel-wise predictions (a) yield candidates through merge-guided Gaussian proposal (b). Existence-aware candidate validation suppresses a candidate in (c), while opacity aging adjusts the rendering contributions of Gaussians.

produce a feature map $\mathbf{F}_t$. We retain the causal extrinsic head $\Phi_{\text{ext}}$ of OF$^3$GS Chen et al. (2026) to predict rotation $\hat{\mathbf{R}}_t$ and translation $\hat{\mathbf{t}}_t$ from $\mathbf{F}_t$. A frozen monocular intrinsic head $\Phi_{\text{int}}$ (MoGe-2 Wang et al. (2025)) estimates the intrinsic matrix $\hat{\mathbf{K}}$ from the first frame $\mathbf{I}_0$. This matrix is kept fixed and shared across all frames in the video stream. We use a Gaussian decoder $\mathcal{D}$ to map $\mathbf{F}_t$ to depth $\mathbf{D}_t$, per-pixel Gaussian attributes $\mathbf{A}_t$ (rotation, scale, opacity, and color), a merge map $\mathbf{H}_t$, and an existence map $\mathbf{M}_t$. Depth and attributes provide dense Gaussian predictions, while the two maps guide candidate proposal and validation below.

### 4.1 Gaussian Lifecycle Modeling

Dense per-frame Gaussian prediction can repeatedly represent the same scene regions, introducing redundant primitives that may degrade rendering quality. We use $\mathbf{H}_t$ to guide candidate placement and $\mathbf{M}_t$ to determine candidate retention in $\mathcal{G}_t$. Together, the maps control the placement and number of newly added Gaussians. Since a Gaussian's reliability can vary across rendering views, opacity aging adjusts its rendering contribution by view and time. The three stages are described below.

#### 4.1.1 Merge-Guided Gaussian Proposal

We use the merge map $\mathbf{H}_t$ to predict candidate positions within non-overlapping $7\times7$ image patches, as illustrated by the transition from Fig. 3(a) to (b). Let $\Omega_i$ denote patch $i$. The merge map has $C_H$ channels, each determining one candidate location per patch, indexed by the patch index $i$ and channel index $c$.

For pixel $\mathbf{u} \in \Omega_i$, let $\mathbf{H}_t^c[\mathbf{u}]$ denote its predicted score and $\mathbf{y}(\mathbf{u})$ its 2D pixel coordinate. We normalize the scores into weights $p_{i,c}(\mathbf{u})$ and compute the candidate position $\mathbf{y}_{i,c}$:

$$p_{i,c}(\mathbf{u}) = \frac{\exp(\mathbf{H}_t^c[\mathbf{u}])}{\sum_{\mathbf{v}\in\Omega_i}\exp(\mathbf{H}_t^c[\mathbf{v}])}, \qquad \mathbf{y}_{i,c} = \sum_{\mathbf{u}\in\Omega_i} p_{i,c}(\mathbf{u})\,\mathbf{y}(\mathbf{u}). \tag{2}$$

Here, $\mathbf{v}$ indexes pixels in the same patch. The weighted average allows each channel to choose a position according to the predicted scores.

We round $\mathbf{y}_{i,c}$ to the nearest pixel $\mathbf{u}_{i,c}$ and sample depth $\mathbf{D}_t[\mathbf{u}_{i,c}]$ and attributes $\mathbf{A}_t[\mathbf{u}_{i,c}]$. Let $\tilde{\mathbf{u}}_{i,c}$ denote the homogeneous image coordinates of the selected pixel $\mathbf{u}_{i,c}$. We back-project $\mathbf{u}_{i,c}$ to obtain the candidate center $\boldsymbol{\mu}_{i,c}$ in world coordinates:

$$\boldsymbol{\mu}_{i,c} = \hat{\mathbf{R}}_t^{-1}\left(\mathbf{D}_t[\mathbf{u}_{i,c}]\hat{\mathbf{K}}^{-1}\tilde{\mathbf{u}}_{i,c} - \hat{\mathbf{t}}_t\right). \tag{3}$$

The selected candidate Gaussian $\mathbf{g}_{i,c}$ has the back-projected center $\boldsymbol{\mu}_{i,c}$ and sampled attributes $\mathbf{A}_t[\mathbf{u}_{i,c}]$, and is then passed to existence-aware validation. During training, straight-through rounding and differentiable grid sampling allow rendering supervision to train $\mathbf{H}_t$; implementation details are provided in the supplementary material.

#### 4.1.2 EXISTENCE-AWARE CANDIDATE VALIDATION

As shown in Fig. 3(b)–(c), existence-aware candidate validation suppresses redundant candidates before accumulation. The existence map $\mathbf{M}_t$ has the same $C_H$ channels as $\mathbf{H}_t$. For each selected candidate Gaussian $\mathbf{g}_{i,c}$, we obtain its existence logit $\mathbf{M}_t^c[\mathbf{u}_{i,c}]$ from channel $c$.

At inference, we convert this logit into a soft existence confidence $m_{i,c}^{\text{soft}}$ and keep candidates whose confidence exceeds $0.5$:

$$\begin{aligned} m_{i,c}^{\text{soft}} &= \sigma(\mathbf{M}_t^c[\mathbf{u}_{i,c}]/\tau)\,, \\ \mathcal{G}_t &= \{\mathbf{g}_{i,c} \mid m_{i,c}^{\text{soft}} > 0.5\}. \end{aligned} \tag{4}$$

Here, $\sigma$ is the sigmoid function and $\tau > 0$ is its temperature. During training, binary gates multiply the candidate opacities in $\mathbf{A}_t$, so inactive candidates have zero rendering contribution but remain in $\mathcal{G}_t$ with gated opacity. A noisy continuous relaxation and a straight-through estimator allow rendering supervision to train these decisions. At inference, inactive candidates are discarded. The training gate is detailed in the supplementary material.

The current frame's output $\mathcal{G}_t$ is appended to $\mathcal{G}_{\leq t}$ as in Eq. (1). Each Gaussian $\mathbf{g}_i \in \mathcal{G}_{\leq t}$ stores its source frame $s(i)$ and soft existence confidence $\eta_i$, inherited from its corresponding candidate. These support opacity aging (Sec. 4.1.3) and chunk-wise streaming (Sec. 4.2), respectively.

#### 4.1.3 VIEW- AND TIME-AWARE OPACITY AGING

To favor observations close to the requested view, we weight Gaussians in $\mathcal{G}_{\leq t}$ by source viewing direction and time, as illustrated in Fig. 3(c). For long videos, we apply the weights to the merged global Gaussian set from Sec. 4.2. Aging changes rendering contributions without modifying stored Gaussian attributes. Here, $i$ indexes Gaussians, $\alpha_i$ is Gaussian $i$'s post-validation opacity (including the gate during training), and $r$ denotes the rendering view.

**View decay.** Let $\theta_{r,i}$ be the angle between the directions from Gaussian $i$ toward its source camera and toward the rendering camera. We define the view-dependent weight $w_{r,i}^v$ as

$$w_{r,i}^v = \exp[-\lambda_v(1 - \cos\theta_{r,i})]\,, \tag{5}$$

where $\lambda_v \geq 0$ controls decay strength. The weight decreases as the source and rendering directions become less consistent.

**Temporal decay.** We select an input frame $j(r)$ as the temporal reference for rendering view $r$. For a context view, $j(r) = r$; for a target view, we select the already processed frame closest in camera position and orientation, using the distance measure given in the supplementary material. Let $\Delta t_{r,i} = |\bar{t}(j(r)) - \bar{t}(s(i))|$, where $\bar{t}(a)$ is the timestamp of input frame $a$ normalized within the processed sequence. The temporal weight $w_{r,i}^t$ is

$$w_{r,i}^t = \exp(-\lambda_t \Delta t_{r,i})\,\chi_{r,i}, \tag{6}$$

where $\lambda_t \geq 0$ controls the decay rate and $\chi_{r,i}$ is a binary temporal mask. During training, $\chi_{r,i} = 1$ for all Gaussians. At inference, it equals one only when the source frame $s(i)$ is within $R$ chronological frame indices of $j(r)$, and zero otherwise; $R$ is the temporal window radius.

**Final opacity.** The final opacity $\alpha_{r,i}^{\text{final}}$ combines the post-validation opacity with both weights:

$$\alpha_{r,i}^{\text{final}} = \alpha_i\, w_{r,i}^v\, w_{r,i}^t. \tag{7}$$

Rendering uses this opacity with the Gaussian geometry and colors. During training, the rendered images provide photometric supervision (Sec. 4.3).

### 4.2 CHUNK-WISE STREAMING FOR LONG VIDEOS

To extend our framework to long videos, we reconstruct chunks of $C$ frames with $O$ overlapping frames, where $0 < O < C$. Each chunk resets its feature-context state and local Gaussian set, then proposes and validates Gaussians using Secs. 4.1.1–4.1.2. All chunks share the intrinsics estimated from the first frame of the entire video. Each chunk outputs local Gaussians, camera estimates, and soft existence confidences.

We align neighboring chunks through their shared frames. For each shared frame, we back-project predicted depths using the common intrinsics and chunk-specific extrinsics. Points at the same pixel of the same frame form a correspondence. Let $\mathcal{M}_k$ be the correspondence set between chunks $k$ and $k-1$. For match $\ell$, $\mathbf{p}_\ell^k$ is the point in chunk $k$, and $\bar{\mathbf{p}}_\ell^{k-1}$ is the previous chunk's point expressed in global coordinates. Each selected pixel carries its candidate's scalar soft existence confidence from the corresponding existence-map channel (Sec. 4.1.2). We use these confidences to weight the matched points. Let $\eta_\ell^k$ and $\bar{\eta}_\ell^{k-1}$ denote the confidences attached to the matched points from chunks $k$ and $k-1$, respectively. We assign the match weight $\omega_\ell = \eta_\ell^k \bar{\eta}_\ell^{k-1}$. Following VGGT-Long Deng et al. (2025), we estimate a similarity transform $\mathbf{S}_k = (s_k, \mathbf{R}_k, \mathbf{t}_k)$ by robust alignment:

$$\mathbf{S}_k = \underset{s>0,\, \mathbf{R}\in \mathrm{SO}(3),\, \mathbf{t}}{\operatorname{argmin}} \sum_{\ell \in \mathcal{M}_k} \omega_\ell \, \rho\big(\big\| s\mathbf{R}\mathbf{p}_\ell^k + \mathbf{t} - \bar{\mathbf{p}}_\ell^{k-1} \big\|_2\big) \,. \tag{8}$$

Here, $s$, $\mathbf{R}$, and $\mathbf{t}$ are scale, rotation, and translation, respectively, and $\rho$ is the Huber penalty.

We transform the chunk's cameras and Gaussians into global coordinates, adjusting Gaussian centers, orientations, and scales while leaving other attributes unchanged. For duplicate Gaussians associated with the same pixel of a shared frame, we keep the copy with the higher soft existence confidence. Gaussians from new frames are appended with their source-frame information. The merged set and aligned cameras support opacity aging and rendering in Sec. 4.1.3. Processing fixed-length chunks limits the memory required for historical feature context, while the accumulated Gaussian set can still grow as the video continues.

### 4.3 Training Objective

Following OF$^3$GS Chen et al. (2026), we optimize $\mathcal{L} = \mathcal{L}_{\text{photo}} + \lambda_{\text{depth}} \mathcal{L}_{\text{depth}}$, where $\lambda_{\text{depth}}$ is the depth-loss weight. The photometric term combines MSE, SSIM, and LPIPS over context and held-out novel views, with greater weight on novel views. The depth term aligns predicted depth to a Depth Anything 3 Lin et al. (2025b) prior by a per-frame scale and shift before applying an $\ell_2$ loss. Together, the losses stabilize lifting and improve rendering from unseen viewpoints.

## 5 Experiments

### 5.1 Experimental Setup

**Baselines and data.** We compare LifeStreamGS with the online feed-forward baseline OF$^3$GS Chen et al. (2026), online optimization-based methods (OnTheFly-NVS Meuleman et al. (2025) and ART-DECO Li et al. (2025a)), and offline feed-forward baselines (AnySplat Jiang et al. (2025) and World-Mirror Liu et al. (2025)).

LifeStreamGS is trained on $10,000$ sequences from DL3DV Ling et al. (2024) and $10,000$ trajectories from RE10K Zhou et al. (2018). We evaluate in-domain NVS on DL3DV-140 Ling et al. (2024) and 200 RE10K Zhou et al. (2018) test scenes, and zero-shot generalization on unseen NYUv2 Silberman et al. (2012) scenes. We evaluate long-sequence reconstruction on 500-view streams from 7-Scenes Shotton et al. (2013) and Tanks & Temples Knapitsch et al. (2017). We report PSNR, SSIM, and LPIPS as evaluation metrics.

**Implementation.** During training, we randomly sample video clips containing between 2 and 16 frames and optimize LifeStreamGS for five epochs on four NVIDIA RTX 5880 Ada GPUs. Images are resized to a width of 518 pixels while preserving aspect ratio. We use a batch size of 1 per GPU, giving a global batch size of 4. Following OF$^3$GS Chen et al. (2026), the causal feature extractor and extrinsic head are initialized from StreamVGGT Zhuo et al. (2025) and kept frozen; the MoGe-2 Wang et al. (2025) intrinsic estimator and Depth Anything 3 Lin et al. (2025b) prior are also frozen. We use AdamW Loshchilov & Hutter (2017) with learning rates of $5 \times 10^{-5}$ for parameters initialized from scratch and $1 \times 10^{-5}$ for pretrained parameters, weight decay $0.1$, and a $1{,}000$-step warm-up. The merge and existence maps each use $C_H = 32$ channels. For long-sequence inference, we use chunks of $C = 30$ frames with overlap $O = 15$. Further training and inference settings are provided in the supplementary material.

| Dataset | Method | Online? | Input Views: 5 | | | | | Input Views: 64 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | PSNR ↑ | SSIM ↑ | LPIPS ↓ | Time (s) ↓ | Mem. (GB) ↓ | PSNR ↑ | SSIM ↑ | LPIPS ↓ | Time (s) ↓ | Mem. (GB) ↓ |
| *DL3DV -140 Ling et al. (2024)* | AnySplat Jiang et al. (2025) ⋆ | × | 19.128 | 0.565 | 0.391 | 4.51 | 16.23 | OOM | | | | |
| | WorldMirror Liu et al. (2025) ⋆ | × | 21.736 | 0.673 | 0.285 | 3.99 | 15.41 | OOM | | | | |
| | OnTheFly-NVS Meuleman et al. (2025) † | ✓ | 21.030 | 0.644 | 0.325 | **1.96** | **5.05** | 18.478 | 0.610 | 0.455 | 38.98 | 11.54 |
| | ARTDECO Li et al. (2025a) † | ✓ | 21.740 | 0.699 | 0.312 | 16.86 | 6.97 | 21.982 | 0.687 | 0.341 | 62.24 | 7.51 |
| | OF$^3$GS Chen et al. (2026)⋆ | ✓ | 21.836 | 0.678 | 0.287 | 2.68 | 7.51 | 17.190 | 0.558 | 0.485 | 31.65 | 13.93 |
| | **Ours**⋆ | ✓ | **22.515** | **0.700** | **0.266** | 2.13 | 6.09 | **22.151** | **0.701** | **0.264** | **22.47** | **6.11** |
| *RE10K Zhou et al. (2018)* | AnySplat Jiang et al. (2025) ⋆ | × | 21.856 | 0.732 | 0.273 | 3.61 | 14.24 | OOM | | | | |
| | WorldMirror Liu et al. (2025) ⋆ | × | 26.006 | 0.841 | 0.195 | 3.56 | 15.12 | OOM | | | | |
| | OnTheFly-NVS Meuleman et al. (2025) † | ✓ | 17.829 | 0.615 | 0.390 | 2.89 | 6.14 | 18.926 | 0.708 | 0.361 | 37.24 | 11.44 |
| | ARTDECO Li et al. (2025a) † | ✓ | 25.904 | **0.869** | **0.175** | 17.45 | 6.12 | **23.806** | **0.771** | **0.219** | 63.56 | 8.11 |
| | OF$^3$GS Chen et al. (2026)⋆ | ✓ | 25.225 | 0.816 | 0.210 | 3.24 | 7.81 | 19.765 | 0.696 | 0.331 | 30.32 | 12.46 |
| | **Ours**⋆ | ✓ | **26.112** | 0.836 | 0.202 | **2.02** | **5.89** | 22.651 | 0.756 | 0.243 | **22.35** | **6.01** |

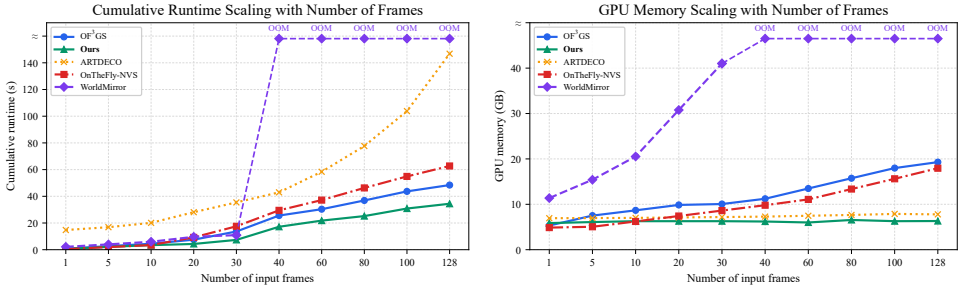


Figure 4: **Top:** Quantitative comparison on DL3DV-140 Ling et al. (2024) and RE10K Zhou et al. (2018) with 5 and 64 input views. Time is cumulative runtime (s); memory is GPU memory (GB). ⋆ denotes feed-forward and † optimization-based methods. Best and second-best results are **bold** and underlined. **Bottom:** cumulative runtime (left) and GPU memory (right) through 128 input frames on DL3DV Ling et al. (2024). OOM denotes out-of-memory failure.

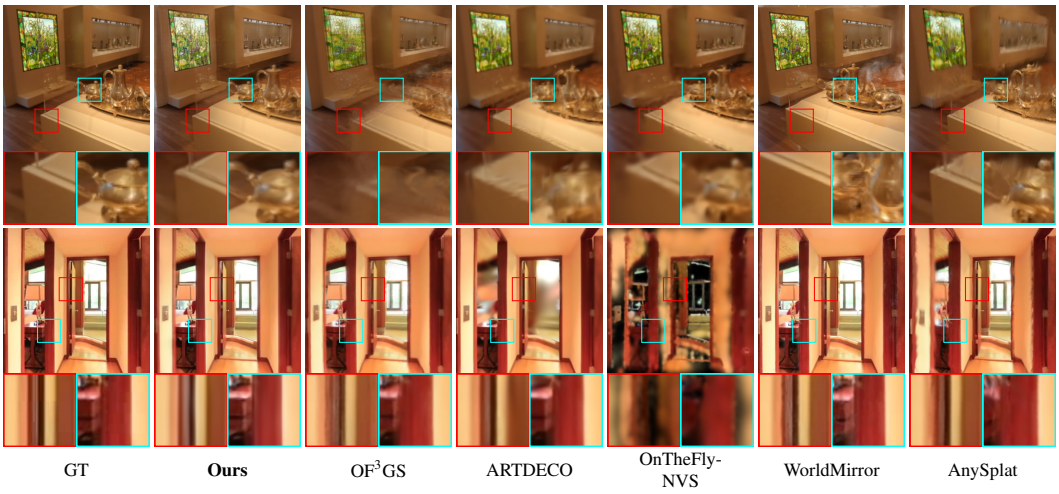


Figure 5: Qualitative comparison with 5 input views on DL3DV-140 Ling et al. (2024) (top) and RE10K Zhou et al. (2018) (bottom). Ours yields cleaner boundaries with fewer artifacts and floaters.

### 5.2 Experimental Results

**Evaluation on novel view synthesis.** The top table in Fig. 4 compares NVS results on DL3DV-140 Ling et al. (2024) and RE10K Zhou et al. (2018) with 5 and 64 input views. LifeStreamGS achieves the best or second-best reconstruction quality among online methods across all evaluated settings. The time and memory columns show that LifeStreamGS is 2.8–8.6× faster than ARTDECO Li et al. (2025a) and uses less GPU memory in every setting.

Table 1: Quantitative comparison under 500 input views.

| Method | **7-Scenes** (500 Views) | | | **Tanks & Temples** (500 Views) | | |
|---|---|---|---|---|---|---|
| | PSNR ↑ | SSIM ↑ | LPIPS ↓ | PSNR ↑ | SSIM ↑ | LPIPS ↓ |
| OnTheFly-NVS Meuleman et al. (2025) † | 19.880 | 0.699 | 0.423 | 19.205 | 0.665 | 0.388 |
| ARTDECO Li et al. (2025a) † | <u>23.134</u> | **0.792** | <u>0.353</u> | <u>20.954</u> | **0.694** | <u>0.372</u> |
| OF$^3$GS Chen et al. (2026)⋆ | —— | OOM | —— | —— | OOM | —— |
| LifeStreamGS w/o chunking⋆ | —— | OOM | —— | —— | OOM | —— |
| **Ours**⋆ | **23.997** | <u>0.768</u> | **0.272** | **21.509** | <u>0.678</u> | **0.269** |

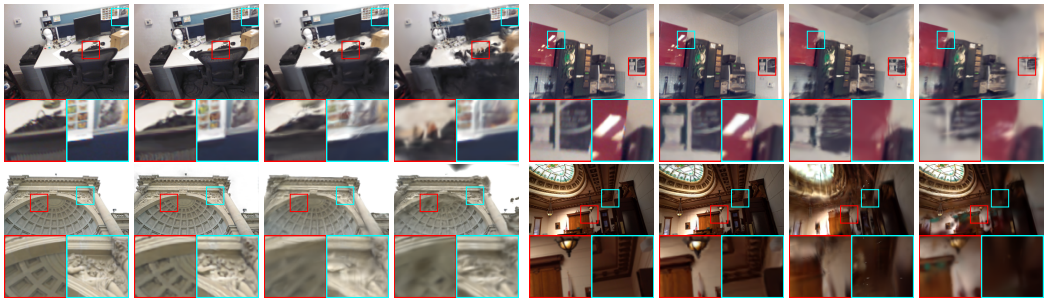


Figure 6: Qualitative comparison with 500 input views on 7-Scenes (top) and Tanks & Temples (bottom). Ours produces sharper details with fewer blurred or ghosted regions.

Table 2: Quantitative zero-shot comparison on NYUv2 Silberman et al. (2012) with 5/64 inputs.

| Method | Online? | Input Views: 5 | | | | | Input Views: 64 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | PSNR ↑ | SSIM ↑ | LPIPS ↓ | Time (s) ↓ | Mem. (GB) ↓ | PSNR ↑ | SSIM ↑ | LPIPS ↓ | Time (s) ↓ | Mem. (GB) ↓ |
| AnySplat Jiang et al. (2025) ⋆ | × | 22.047 | 0.648 | 0.319 | 4.04 | 15.12 | OOM | | | | |
| WorldMirror Liu et al. (2025) ⋆ | × | 24.750 | 0.713 | 0.303 | 3.64 | 15.21 | OOM | | | | |
| OnTheFly-NVS Meuleman et al. (2025) † | ✓ | 20.875 | 0.635 | 0.376 | <u>2.54</u> | <u>6.85</u> | 23.174 | 0.679 | 0.352 | 39.64 | 12.12 |
| ARTDECO Li et al. (2025a) † | ✓ | <u>25.262</u> | <u>0.718</u> | 0.338 | 18.23 | 7.51 | **25.195** | **0.785** | **0.292** | 64.39 | <u>8.89</u> |
| OF$^3$GS Chen et al. (2026)⋆ | ✓ | 24.875 | 0.717 | **0.291** | 3.24 | 7.03 | 21.253 | 0.652 | 0.361 | <u>33.23</u> | 14.05 |
| **Ours**⋆ | ✓ | **25.267** | **0.747** | <u>0.299</u> | **2.05** | **6.31** | <u>23.933</u> | <u>0.705</u> | <u>0.308</u> | **21.35** | **6.03** |

The runtime and memory curves at the bottom of Fig. 4 show how computational requirements change as the input sequence grows. LifeStreamGS processes all 128 frames with lower cumulative runtime than the online baselines, while keeping memory usage near 6 GB. The offline baseline WorldMirror Liu et al. (2025) runs out of memory as more views are added. Together, the quality rankings and runtime–memory results demonstrate a strong balance of rendering quality, speed, and memory usage for online NVS.

In Fig. 5, LifeStreamGS preserves clearer object boundaries and finer scene details, with fewer artifacts in the highlighted regions. ARTDECO Li et al. (2025a) produces blurrier boundaries and fewer visible details in these examples. These comparisons show that LifeStreamGS preserves scene structure and visual detail in online reconstruction.

**Long-sequence evaluation.** Table 1 reports reconstruction results from 500 input views on 7-Scenes Shotton et al. (2013) and Tanks & Temples Knapitsch et al. (2017). LifeStreamGS achieves the best or second-best quality across both datasets. Compared with ARTDECO Li et al. (2025a), LifeStreamGS achieves better results on most metrics in both evaluation settings. Without chunk-wise streaming, LifeStreamGS runs out of memory in this setting, while the full LifeStreamGS model completes reconstruction. In the highlighted regions of Fig. 6, LifeStreamGS reconstructs sharper scene details with fewer blurred or ghosted regions than ARTDECO Li et al. (2025a). These results support combining Gaussian lifecycle modeling with chunk-wise streaming to maintain reconstruction quality over long videos.

**Cross-dataset generalization.** Table 2 reports zero-shot NVS results on unseen NYUv2 Silberman et al. (2012) scenes without fine-tuning. LifeStreamGS achieves the best or second-best reconstruction quality across both input settings. Compared with ARTDECO Li et al. (2025a), LifeStreamGS provides competitive quality with a 3.0–8.9× speedup and lower GPU memory usage. These results

Table 3: Ablation study of LifeStreamGS under 5 input views. Checkmarks indicate enabled lifecycle components.

| Lifecycle Components | | | DL3DV | | | | RE10K | | | | NYUv2 (Gen.) | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Merge-guided GS Proposal | Exist.-aware Can. Val. | Opacity Aging | PSNR ↑ | SSIM ↑ | LPIPS ↓ | # GS ↓ | PSNR ↑ | SSIM ↑ | LPIPS ↓ | # GS ↓ | PSNR ↑ | SSIM ↑ | LPIPS ↓ | # GS ↓ |
| × | × | × | 21.446 | 0.643 | 0.347 | 1341620 | 24.674 | 0.775 | 0.241 | 1341620 | 24.634 | 0.720 | 0.333 | 1341620 |
| × | ✓ | ✓ | 22.270 | 0.678 | 0.283 | 931729 | 25.924 | 0.818 | 0.235 | 875676 | 24.975 | 0.711 | 0.313 | 931864 |
| ✓ | × | ✓ | 22.303 | 0.655 | 0.278 | 876160 | 25.825 | 0.811 | 0.217 | 876160 | 24.908 | 0.723 | 0.310 | 876160 |
| ✓ | ✓ | × | 21.671 | 0.674 | 0.320 | **247262** | 24.896 | 0.809 | 0.236 | **243905** | 24.847 | 0.737 | 0.322 | **246657** |
| ✓ | ✓ | ✓ | **22.515** | **0.700** | **0.266** | 247262 | **26.112** | **0.836** | **0.202** | 243905 | **25.267** | **0.747** | **0.299** | 246657 |

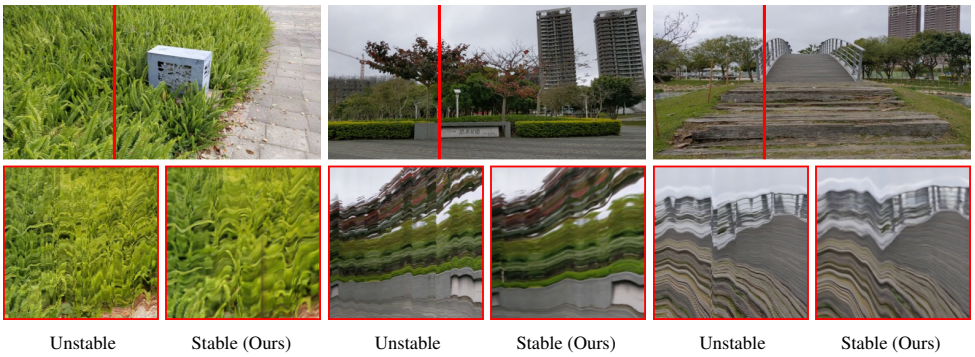


Figure 7: Online video stabilization. **Top**: sampled columns. **Bottom**: X-T temporal profiles. More results are provided in the supplementary videos.

demonstrate generalization to unseen scenes while retaining a favorable balance of quality, speed, and memory usage. Qualitative comparisons are provided in the supplementary material.

**Ablation study.** Table 3 shows that removing the full lifecycle model causes the largest overall quality degradation. Removing any module individually also lowers reconstruction quality across the evaluated datasets. These results confirm that the three components provide complementary benefits and jointly improve online NVS. Additional controlled comparisons and qualitative ablations are included in the supplementary material.

**Application: online video stabilization.** We further apply LifeStreamGS to online video stabilization. Specifically, we smooth the estimated camera extrinsic trajectory using a Savitzky–Golay filter Savitzky & Golay (1964) and perform NVS along the smoothed trajectory using the incrementally reconstructed Gaussian set. Figure 7 visualizes the temporal evolution through X-T profiles, formed by concatenating the same image column across successive frames. The smoother profiles show reduced high-frequency camera shake and demonstrate stable online video rendering. Additional dynamic results are provided in the supplementary videos.

## 6 Conclusion

We presented LifeStreamGS, a streaming feed-forward framework for 3DGS reconstruction from unposed videos. Gaussian lifecycle modeling combines merge-guided Gaussian proposal, existence-aware candidate validation, and view- and time-aware opacity aging to reduce redundant accumulation and limit the impact of prediction errors on rendering. Chunk-wise streaming extends the framework to long videos by reconstructing overlapping windows and aligning their reconstructions into a shared coordinate system. Experiments demonstrate competitive NVS quality with favorable runtime and GPU memory usage, including reconstruction from 500-view streams and an application to online video stabilization.

**Limitations.** LifeStreamGS assumes mostly static scenes and fixed camera intrinsics, so rapid non-rigid motion and focal-length changes remain challenging. The accumulated Gaussian set can still grow over long videos and may require further compression.

## AI USE STATEMENT

Generative AI tools were used for language editing, LaTeX formatting, figure preparation, reference checking, and discussions to clarify the method formulation and interpret author-provided experimental results. They were not used to generate experimental data. All AI-assisted text, code, figures, and analyses were reviewed and verified by the authors, who take full responsibility for the final content of this work.

## REFERENCES

Minh-Quan Viet Bui, Jaeho Moon, and Munchurl Kim. AirSplat: Alignment and rating for robust feed-forward 3D gaussian splatting. In *ECCV*, 2026.

David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelSplat: 3D gaussian splats from image pairs for scalable generalizable 3D reconstruction. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 19457–19467, 2024.

Ruiyang Chen, Feiran Li, Chu Zhou, Zonglin Li, Zhanyu Ma, and Heng Guo. OF3GS: On-the-fly feed-forward 3D gaussian splatting from unposed images. *arXiv preprint arXiv:2606.03254*, 2026.

Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. MVSplat: Efficient 3D gaussian splatting from sparse multi-view images. In *European Conference on Computer Vision*, pp. 370–386. Springer, 2024.

Freeman Cheng, Botao Ye, Xueting Li, Junqi You, Fangneng Zhan, and Ming-Hsuan Yang. ReCoSplat: Online feed-forward gaussian splatting via render-and-compare. *arXiv preprint arXiv:2603.09968*, 2026.

Kai Deng, Zexin Ti, Jiawei Xu, Jian Yang, and Jin Xie. VGGT-Long: Chunk it, loop it, align it–pushing VGGT's limits on kilometer-scale long RGB sequences. *arXiv preprint arXiv:2507.16443*, 2025.

Jiaxin Guo, Tongfan Guan, Wenzhen Dong, Wenzhao Zheng, Wenting Wang, Yue Wang, Yeung Yam, and Yun-Hui Liu. SaLon3R: Structure-aware long-term generalizable 3D reconstruction from unposed images. *arXiv preprint arXiv:2510.15072*, 2025.

Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2D gaussian splatting for geometrically accurate radiance fields. In *SIGGRAPH 2024 Conference Papers*, pp. 1–11. Association for Computing Machinery, 2024. doi: 10.1145/3641519.3657428.

Roni Itkin, Noam Issachar, Yehonatan Keypur, Xingyu Chen, Anpei Chen, and Sagie Benaim. GlobalSplat: Efficient feed-forward 3D gaussian splatting via global scene tokens. *arXiv preprint arXiv:2604.15284*, 2026.

Lihan Jiang, Yucheng Mao, Linning Xu, Tao Lu, Kerui Ren, Yichen Jin, Xudong Xu, Mulin Yu, Jiangmiao Pang, Feng Zhao, Dahua Lin, and Bo Dai. AnySplat: Feed-forward 3D gaussian splatting from unconstrained views. *ACM Transactions on Graphics (TOG)*, 44(6):1–16, 2025.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):1–14, 2023.

Injae Kim, Chaehyeon Kim, Minseong Bae, Minseok Joo, and Hyunwoo J. Kim. F4Splat: Feed-forward predictive densification for feed-forward 3D gaussian splatting. *arXiv preprint arXiv:2603.21304*, 2026.

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics*, 36(4):1–13, 2017.

Guanghao Li, Kerui Ren, Linning Xu, Zhewen Zheng, Changjian Jiang, Xin Gao, Bo Dai, Jian Pu, Mulin Yu, and Jiangmiao Pang. ARTDECO: Towards efficient and high-fidelity on-the-fly 3D reconstruction with structured scene representation. *arXiv preprint arXiv:2510.08551*, 2025a.

Yang Li, Jinglu Wang, Lei Chu, Xiao Li, Shiu-hong Kao, Ying-Cong Chen, and Yan Lu. StreamGS: Online generalizable gaussian splatting reconstruction for unposed image streams. *arXiv preprint arXiv:2503.06235*, 2025b.

Chin-Yang Lin, Cheng Sun, Fu-En Yang, Min-Hung Chen, Yen-Yu Lin, and Yu-Lun Liu. LongSplat: Robust unposed 3D gaussian splatting for casual long videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 27412–27422, 2025a.

Haotong Lin, Sili Chen, Jun Hao Liew, Donny Y. Chen, Zhenyu Li, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth Anything 3: Recovering the visual space from any views. *arXiv preprint arXiv:2511.10647*, 2025b.

Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, Xuanmao Li, Xingpeng Sun, Rohan Ashok, Aniruddha Mukherjee, Hao Kang, Xiangrui Kong, Gang Hua, Tianyi Zhang, Bedrich Benes, and Aniket Bera. DL3DV-10K: A large-scale scene dataset for deep learning-based 3D vision. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 22160–22169, 2024.

Yifan Liu, Zhiyuan Min, Zhenwei Wang, Junta Wu, Tengfei Wang, Yixuan Yuan, Yawei Luo, and Chunchao Guo. WorldMirror: Universal 3D world reconstruction with any-prior prompting. *arXiv preprint arXiv:2510.10726*, 2025.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Andreas Meuleman, Ishaan Shah, Alexandre Lanvin, Bernhard Kerbl, and George Drettakis. On-the-fly reconstruction for large-scale novel view synthesis from unposed images. *ACM Transactions on Graphics (TOG)*, 44(4):1–14, 2025.

Jiawei Ren, Michal Jan Tyszkiewicz, Jiahui Huang, and Zan Gojcic. TokenGS: Decoupling 3D gaussian prediction from pixels with learnable tokens. *arXiv preprint arXiv:2604.15239*, 2026.

Abraham Savitzky and Marcel J. E. Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry*, 36(8):1627–1639, 1964.

Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in RGB-D images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2930–2937, 2013.

Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from RGBD images. In *European Conference on Computer Vision*, pp. 746–760, 2012.

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

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004.

Zike Wu, Qi Yan, Xuanyu Yi, Lele Wang, and Renjie Liao. StreamSplat: Towards online dynamic 3D reconstruction from uncalibrated video streams. *arXiv preprint arXiv:2506.08862*, 2025.

Yiwei Xu, Yifei Yu, Wentian Gan, Tengfei Wang, Zongqian Zhan, Hao Cheng, and Xin Wang. Gaussian on-the-fly splatting: A progressive framework for robust near real-time 3DGS optimization. *arXiv preprint arXiv:2503.13086*, 2025.

Botao Ye, Sifei Liu, Haofei Xu, Xueting Li, Marc Pollefeys, Ming-Hsuan Yang, and Songyou Peng. No pose, no problem: Surprisingly simple 3D gaussian splats from sparse unposed images. *arXiv preprint arXiv:2410.24207*, 2024.

Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-Splatting: Alias-free 3D gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19447–19456, 2024.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018.

Shangzhan Zhang, Jianyuan Wang, Yinghao Xu, Nan Xue, Christian Rupprecht, Xiaowei Zhou, Yujun Shen, and Gordon Wetzstein. FLARE: Feed-forward geometry, appearance and camera estimation from uncalibrated sparse views. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 21936–21947, 2025.

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

Dong Zhuo, Wenzhao Zheng, Jiahe Guo, Yuqi Wu, Jie Zhou, and Jiwen Lu. Streaming 4D visual geometry transformer. *arXiv preprint arXiv:2507.11539*, 2025.

# SUPPLEMENTARY MATERIAL FOR LIFESTREAMGS

## OVERVIEW

In this supplementary material, we provide additional details organized as follows. Please see the supplementary videos for demonstrations of online feed-forward 3DGS reconstruction and NVS for long videos and online video stabilization.


- Section 1: Additional method analysis.
  - Section 1.1: Detailed method formulation.
- Section 2: Additional experimental results.
  - Section 2.1: Experimental protocol details.
  - Section 2.2: Additional NVS results.
  - Section 2.3: Additional long-sequence results.
  - Section 2.4: Additional cross-dataset results.
  - Section 2.5: Additional ablation results.


## 1 ADDITIONAL METHOD ANALYSIS

### 1.1 DETAILED METHOD FORMULATION

**Details of Merge-Guided Gaussian Proposal.** Section 4.1.1 of the main paper defines a continuous candidate location $\mathbf{y}_{i,c}$ for patch $i$ and merge-map channel $c$. We implement rounding with a straight-through coordinate:

$$\bar{\mathbf{y}}_{i,c} = \mathrm{sg}(\mathrm{round}(\mathbf{y}_{i,c}) - \mathbf{y}_{i,c}) + \mathbf{y}_{i,c}. \tag{1}$$

Here, $\mathrm{round}(\cdot)$ rounds each coordinate to the nearest integer pixel location and $\mathrm{sg}(\cdot)$ stops gradients. Thus, the forward coordinate is rounded, while its gradient with respect to $\mathbf{y}_{i,c}$ is the identity. The index operator $\pi$ returns the selected pixel index $\mathbf{u}_{i,c} = \pi(\bar{\mathbf{y}}_{i,c})$, whose homogeneous coordinate is $\tilde{\mathbf{u}}_{i,c} = [u^x_{i,c}, u^y_{i,c}, 1]^\top$.

We sample the depth map $\mathbf{D}_t$ and attribute map $\mathbf{A}_t$ with bilinear grid sampling at $\bar{\mathbf{y}}_{i,c}$. At the rounded forward coordinate, the samples equal the values at $\mathbf{u}_{i,c}$ used by the back-projection equation. Coordinate gradients pass through the grid sampler and straight-through coordinate to the merge map $\mathbf{H}_t$; the discrete index operator is not the gradient path.

**Details of Existence-Aware Candidate Validation.** For each selected candidate Gaussian $\mathbf{g}_{i,c}$, we use the matching existence-map logit $\mathbf{M}^c_t[\mathbf{u}_{i,c}]$ to define

$$\begin{aligned} m^{\mathrm{soft}}_{i,c} &= \sigma((\mathbf{M}^c_t[\mathbf{u}_{i,c}] + \epsilon)/\tau)\,, \\ m^{\mathrm{hard}}_{i,c} &= \delta(m^{\mathrm{soft}}_{i,c} > 0.5), \\ m_{i,c} &= m^{\mathrm{hard}}_{i,c} + m^{\mathrm{soft}}_{i,c} - \mathrm{sg}(m^{\mathrm{soft}}_{i,c}). \end{aligned} \tag{2}$$

Here, $\sigma(z) = 1/(1+\exp(-z))$ is the sigmoid, $\epsilon$ is Gumbel noise during training and zero at inference, and $\tau > 0$ is the temperature. The indicator $\delta(\cdot)$ equals one when its condition holds and zero otherwise. The soft confidence $m^{\mathrm{soft}}_{i,c}$ supplies gradients, $m^{\mathrm{hard}}_{i,c}$ is the binary decision, and $m_{i,c}$ uses this binary value in the forward pass while preserving the soft confidence's gradient.

The effective training opacity is $\alpha^{\mathrm{gate}}_{i,c} = m_{i,c}\alpha_{i,c}$, where $\alpha_{i,c}$ is the opacity component of $\mathbf{A}_t[\mathbf{u}_{i,c}]$. All candidates selected by merge-guided Gaussian proposal remain in the current frame's output $\mathcal{G}_t$ during training, with inactive candidates assigned zero opacity; their gates continue to apply after accumulation in $\mathcal{G}_{\le t}$. At inference, inactive candidates are discarded. Each accumulated Gaussian keeps its source frame $s(i)$ and the corresponding soft existence confidence $\eta_i = m^{\mathrm{soft}}_{i,c}$. The post-validation opacity $\alpha_i$ used for aging includes the gate during training.

**Details of View- and Time-Aware Opacity Aging.** Section 4.1.3 of the main paper defines opacity aging for Gaussian $i$ and rendering view $r$. We first specify the viewing angle $\theta_{r,i}$. Let $\boldsymbol{\mu}_i$ be

the Gaussian center, $s(i)$ its source frame, and $\mathbf{o}_a = -\hat{\mathbf{R}}_a^{-1}\hat{\mathbf{t}}_a$ the camera center of view $a$. The normalized source and rendering directions are

$$\mathbf{v}_{s(i),i} = \frac{\mathbf{o}_{s(i)} - \boldsymbol{\mu}_i}{\|\mathbf{o}_{s(i)} - \boldsymbol{\mu}_i\|}, \qquad \mathbf{v}_{r,i} = \frac{\mathbf{o}_r - \boldsymbol{\mu}_i}{\|\mathbf{o}_r - \boldsymbol{\mu}_i\|}. \tag{3}$$

We compute $\cos\theta_{r,i} = \langle \mathbf{v}_{s(i),i}, \mathbf{v}_{r,i} \rangle$ directly for the view weight, without explicitly evaluating the angle.

For temporal decay, the processed input view $j(r)$ serves as the temporal reference. For a target rendering view, we select this reference using camera-pose proximity. After input frames $0{:}t$ have been processed, we choose $j(r) \in \{0, \ldots, t\}$ by comparing the query camera pose of $r$ with the camera poses of the processed input views:

$$j(r) = \operatorname*{argmin}_{j \in \{0,\ldots,t\}} \left( \omega_p d_{\mathrm{pos}}(\mathbf{o}_r, \mathbf{o}_j) + \omega_o d_{\mathrm{ori}}(\hat{\mathbf{R}}_r, \hat{\mathbf{R}}_j) \right), \tag{4}$$

Here $(\hat{\mathbf{R}}_r, \hat{\mathbf{t}}_r)$ denotes the query pose supplied to the renderer, while $(\hat{\mathbf{R}}_j, \hat{\mathbf{t}}_j)$ is the estimated pose of processed input view $j$. Let $\mathbf{f}_a$ be the normalized camera forward axis, obtained from the third column of the camera-to-world rotation of view $a$. We use

$$d_{\mathrm{pos}}(\mathbf{o}_r, \mathbf{o}_j) = \frac{\|\mathbf{o}_r - \mathbf{o}_j\|_2}{m_{\mathrm{pos}}}, \qquad d_{\mathrm{ori}}(\hat{\mathbf{R}}_r, \hat{\mathbf{R}}_j) = 1 - \langle \mathbf{f}_r, \mathbf{f}_j \rangle, \tag{5}$$

where $m_{\mathrm{pos}}$ is the median of all pairwise distances between the rendering-view and processed-input camera centers considered in the current sequence. Across all experiments, we set $\lambda_v = 70$, $\lambda_t = 3$, $\omega_p = 1$, and $\omega_o = 2$. For a context rendering view, the temporal reference is the view itself, i.e., $j(r) = r$. For a target rendering view, the selection above uses only its query camera pose, obtained through the single-pass causal protocol in Sec. 2.1; its image is not used for Gaussian prediction, fusion, or reconstruction-state updates. The temporal distance is $\Delta t_{r,i} = |\bar{t}(j(r)) - \bar{t}(s(i))|$, where $\bar{t}(a) \in [0, 1]$ is the timestamp of input frame $a$ normalized within the processed sequence. The binary temporal mask in the main paper is

$$\chi_{r,i} = \begin{cases} 1, & \text{during training,} \\ \delta(|\operatorname{rank}(j(r)) - \operatorname{rank}(s(i))| \le R), & \text{at inference.} \end{cases} \tag{6}$$

Here, $\operatorname{rank}(a)$ is the chronological index of input frame $a$, and $R$ is the temporal window radius. Substituting this mask into $w_{r,i}^t = \exp(-\lambda_t \Delta t_{r,i})\chi_{r,i}$ gives soft exponential decay during training and additionally suppresses Gaussians outside the window at inference. The final opacity is $\alpha_{r,i}^{\mathrm{final}} = \alpha_i w_{r,i}^v w_{r,i}^t$, with the post-validation opacity $\alpha_i$ defined above.

**Details of Training Objective.** The reconstruction loss combines MSE, SSIM Wang et al. (2004), and LPIPS Zhang et al. (2018):

$$\begin{aligned} \mathcal{L}_{\mathrm{rec}}(\hat{\mathbf{I}}, \mathbf{I}) = {} & \mathrm{MSE}(\hat{\mathbf{I}}, \mathbf{I}) + \lambda_{\mathrm{ssim}} \big(1 - \mathrm{SSIM}(\hat{\mathbf{I}}, \mathbf{I})\big) \\ & + \lambda_{\mathrm{lpips}} \, \mathrm{LPIPS}(\hat{\mathbf{I}}, \mathbf{I}). \end{aligned} \tag{7}$$

Let $\mathbf{I}_v$ and $\hat{\mathbf{I}}_v$ denote the ground-truth and rendered RGB images for supervised view $v$, respectively. Let $\mathcal{V}_{\mathrm{ctx}}$ contain the processed context views and $\mathcal{V}_{\mathrm{novel}}$ contain the held-out novel views used for supervision. The photometric objective is

$$\begin{aligned} \mathcal{L}_{\mathrm{photo}} = {} & \sum_{v \in \mathcal{V}_{\mathrm{ctx}}} \mathcal{L}_{\mathrm{rec}}(\hat{\mathbf{I}}_v, \mathbf{I}_v) \\ & + \lambda_{\mathrm{novel}} \sum_{v \in \mathcal{V}_{\mathrm{novel}}} \mathcal{L}_{\mathrm{rec}}(\hat{\mathbf{I}}_v, \mathbf{I}_v). \end{aligned} \tag{8}$$

Here $\lambda_{\mathrm{ssim}}$ and $\lambda_{\mathrm{lpips}}$ weight the two perceptual terms, and $\lambda_{\mathrm{novel}}$ controls the relative contribution of held-out novel views. For depth supervision, $\mathbf{D}_t^{\mathrm{prior}}$ denotes the frozen Depth Anything 3 Lin et al. (2025b) prediction for frame $t$. We align the Gaussian decoder's depth map $\mathbf{D}_t$ to this pseudo-depth prior by a per-frame scale $\gamma_t$ and shift $\beta_t$:

$$(\gamma_t, \beta_t) = \operatorname*{argmin}_{\gamma,\beta} \left\| \gamma \mathbf{D}_t + \beta - \mathbf{D}_t^{\mathrm{prior}} \right\|_2^2, \tag{9}$$

Table 1: Additional quantitative comparison on DL3DV-140 Ling et al. (2024), RE10K Zhou et al. (2018), and unseen NYUv2 Silberman et al. (2012) scenes with 10 and 128 input views. $^\star$ and $^\dagger$ denote feed-forward and optimization-based methods, respectively. OOM indicates out-of-memory failure. The best and second-best results are shown in **bold** and underlined, respectively.

| Dataset | Method | Online? | Input Views: 10 | | | Input Views: 128 | | |
|---|---|---|---|---|---|---|---|---|
| | | | PSNR ↑ | SSIM ↑ | LPIPS ↓ | PSNR ↑ | SSIM ↑ | LPIPS ↓ |
| *DL3DV -140* | AnySplat Jiang et al. (2025) $^\star$ | × | 18.145 | 0.553 | 0.431 | | OOM | |
| | WorldMirror Liu et al. (2025) $^\star$ | × | 20.533 | **0.697** | **0.291** | | OOM | |
| | OnTheFly-NVS Meuleman et al. (2025) $^\dagger$ | ✓ | 19.470 | 0.612 | 0.393 | 19.750 | 0.655 | 0.398 |
| | ARTDECO Li et al. (2025a) $^\dagger$ | ✓ | **21.018** | 0.658 | 0.357 | **22.478** | **0.713** | 0.315 |
| | OF$^3$GS Chen et al. (2026)$^\star$ | ✓ | 19.386 | 0.590 | 0.387 | 15.605 | 0.563 | 0.504 |
| | **Ours**$^\star$ | ✓ | 20.478 | 0.619 | 0.327 | 21.787 | 0.685 | **0.272** |
| *RE10K* | AnySplat Jiang et al. (2025) $^\star$ | × | 20.379 | 0.705 | 0.324 | | OOM | |
| | WorldMirror Liu et al. (2025) $^\star$ | × | 25.210 | 0.834 | 0.214 | | OOM | |
| | OnTheFly-NVS Meuleman et al. (2025) $^\dagger$ | ✓ | 17.911 | 0.676 | 0.388 | 19.942 | 0.733 | 0.336 |
| | ARTDECO Li et al. (2025a) $^\dagger$ | ✓ | **25.666** | **0.858** | **0.184** | **23.423** | **0.771** | **0.203** |
| | OF$^3$GS Chen et al. (2026)$^\star$ | ✓ | 22.809 | 0.760 | 0.263 | 17.894 | 0.654 | 0.399 |
| | **Ours**$^\star$ | ✓ | 24.788 | 0.799 | 0.222 | 22.686 | 0.757 | 0.242 |
| *NYUv2* | AnySplat Jiang et al. (2025) $^\star$ | × | 22.024 | 0.660 | 0.334 | | OOM | |
| | WorldMirror Liu et al. (2025) $^\star$ | × | 24.212 | 0.694 | 0.297 | | OOM | |
| | OnTheFly-NVS Meuleman et al. (2025) $^\dagger$ | ✓ | 22.515 | 0.629 | 0.323 | 23.497 | 0.752 | 0.323 |
| | ARTDECO Li et al. (2025a) $^\dagger$ | ✓ | **25.704** | **0.746** | **0.283** | **25.049** | **0.787** | **0.285** |
| | OF$^3$GS Chen et al. (2026)$^\star$ | ✓ | 23.215 | 0.684 | 0.307 | 19.445 | 0.612 | 0.409 |
| | **Ours**$^\star$ | ✓ | 24.616 | 0.724 | 0.308 | 23.931 | 0.702 | 0.309 |

and use

$$\mathcal{L}_{\text{depth}} = \sum_t \left\| \gamma_t \mathbf{D}_t + \beta_t - \mathbf{D}_t^{\text{prior}} \right\|_2^2. \tag{10}$$

The scale-shift alignment stabilizes monocular depth supervision without imposing an absolute metric scale.

## 2 Additional Experiments

### 2.1 Experimental Protocol Details

**Evaluation setup.** We use official baseline implementations and protocols without adding external batching, tiling, or chunking. Long-sequence evaluation uses chronologically selected frames from 46 sequences in 7-Scenes Shotton et al. (2013) and 165 samples in Tanks & Temples Knapitsch et al. (2017).

**Target camera poses.** For unposed NVS, LifeStreamGS and OF$^3$GS Chen et al. (2026) append target images after all context images to estimate query poses in the same coordinate system in one causal pass. Target images cannot affect the preceding context reconstruction and are excluded from Gaussian prediction, fusion, and reconstruction-state updates. The remaining baselines follow the evaluation protocols in OF$^3$GS Chen et al. (2026).

**Training and inference settings.** During training, $50\%$ of each clip's frames serve as target views for photometric supervision. The existence-gate temperature is $\tau = 0.8$. Loss weights are $1.0$ for MSE, $\lambda_{\text{ssim}} = \lambda_{\text{lpips}} = 0.05$, $\lambda_{\text{novel}} = 1.5$, and $\lambda_{\text{depth}} = 1.0$. At inference, the temporal window radius is $R = 1$.

### 2.2 Additional NVS Results

Table 1 compares NVS results with 10 and 128 input views on DL3DV-140 Ling et al. (2024), RE10K Zhou et al. (2018), and unseen NYUv2 Silberman et al. (2012) scenes. LifeStreamGS achieves the second-best quality among online methods in most comparisons. At 128 views, the offline baselines run out of memory, while LifeStreamGS completes reconstruction on all three datasets. The runtime and GPU-memory curves in the main paper provide the corresponding ef-

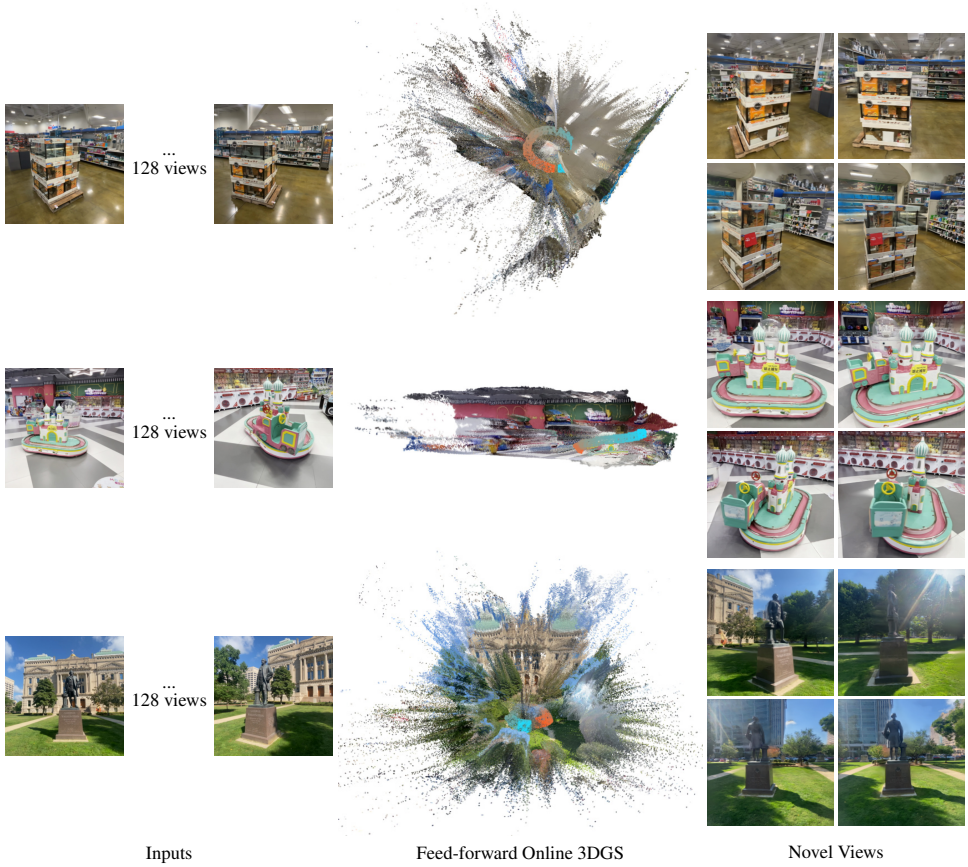


Figure 1: Additional qualitative results on DL3DV-140 Ling et al. (2024) under 128 input views. Each row shows one scene. The first column shows the first and last input views after resize and center crop, the middle column shows the reconstructed 3DGS, and the right column shows rendered novel views from LifeStreamGS.

ficiency comparison. These results support a favorable balance of rendering quality, speed, and memory usage, including generalization to unseen scenes.

Figures 1, 2, and 3 show reconstructions and novel-view renderings from 128 input views on DL3DV-140 Ling et al. (2024), RE10K Zhou et al. (2018), and unseen NYUv2 Silberman et al. (2012) scenes, respectively. LifeStreamGS preserves scene structure and visible object details across these examples. These visual results further support the reconstruction quality and cross-dataset generalization of LifeStreamGS.

### 2.3 Additional Long-Sequence Results

Figures 4 and 5 show reconstructions and novel-view renderings from 500 input views on 7-Scenes Shotton et al. (2013) and Tanks & Temples Knapitsch et al. (2017), respectively. LifeStreamGS preserves scene structure and recognizable details across indoor rooms and larger scenes. These examples show that Gaussian lifecycle modeling and chunk-wise streaming support long-video reconstruction and NVS without per-scene iterative optimization.

### 2.4 Additional Cross-Dataset Results

Figure 6 provides an additional qualitative comparison on unseen NYUv2 Silberman et al. (2012) scenes. Compared with the baseline methods shown in the figure, LifeStreamGS produces clearer object boundaries and fewer visible artifacts in these examples, further supporting its cross-dataset generalization.




Figure 2: Additional qualitative results on RE10K Zhou et al. (2018) under 128 input views. Each row shows one scene. The first column shows the first and last input views after resize and center crop, the middle column shows the reconstructed 3DGS, and the right column shows rendered novel views from LifeStreamGS.

Table 2: Controlled intrinsic replacement under 5 input views. The OF$^3$GS–MoGe-2 variant Chen et al. (2026) uses the same frozen, first-frame MoGe-2 estimator Wang et al. (2025) as LifeStreamGS; all other components follow OF$^3$GS Chen et al. (2026).

| **Method** | **DL3DV** Ling et al. (2024) | | | | **RE10K** Zhou et al. (2018) | | | | **NYUv2 (Gen.)** Silberman et al. (2012) | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | PSNR ↑ | SSIM ↑ | LPIPS ↓ | # GS ↓ | PSNR ↑ | SSIM ↑ | LPIPS ↓ | # GS ↓ | PSNR ↑ | SSIM ↑ | LPIPS ↓ | # GS ↓ |
| OF$^3$GS–MoGe-2 Chen et al. (2026) | 21.728 | 0.668 | 0.289 | 1015739 | 25.235 | 0.826 | 0.201 | 1056829 | 24.871 | 0.715 | 0.293 | 1027262 |
| LifeStreamGS | 22.515 | 0.700 | 0.266 | 247262 | 26.112 | 0.836 | 0.202 | 243905 | 25.267 | 0.747 | 0.299 | 246657 |

## 2.5 Additional Ablation Results

**Controlled intrinsic replacement.** We test whether replacing the intrinsic estimator alone can explain the improvements of LifeStreamGS. The OF$^3$GS–MoGe-2 control Chen et al. (2026) replaces the DIR-Head of OF$^3$GS Chen et al. (2026) with the same frozen, first-frame MoGe-2 estimator Wang et al. (2025) used by LifeStreamGS during both training and inference while keeping all other components of OF$^3$GS Chen et al. (2026) unchanged. Table 2 reports the comparison under 5 input views on DL3DV-140 Ling et al. (2024), RE10K Zhou et al. (2018), and NYUv2 Silberman et al. (2012). The full LifeStreamGS model achieves better overall reconstruction quality than OF$^3$GS–MoGe-2 Chen et al. (2026), showing that replacing the intrinsic estimator alone does not account for the improvements. The third column of Fig. 7 provides the corresponding qualitative results, where boundary and geometry artifacts persist after replacing the intrinsic estimator.

**Lifecycle components.** Removing all lifecycle components blurs the bucket, cup, and chair boundaries and makes local geometry less stable. Without merge-guided Gaussian proposal, layered artifacts appear around object boundaries because candidate locations are no longer adapted to the




Figure 3: Additional qualitative results on NYUv2 Silberman et al. (2012) under 128 input views. Each row shows one scene. The first column shows the first and last input views after resize and center crop, the middle column shows the reconstructed 3DGS, and the right column shows rendered novel views from LifeStreamGS.

image content. Without existence-aware candidate validation, redundant candidates remain active and produce similar boundary-layering artifacts. Without opacity aging, renderings become globally blurrier because view-inconsistent or temporally distant Gaussians remain influential during rendering. Together, these qualitative results support the complementary roles of the three lifecycle components in reducing redundant Gaussian contributions and preserving sharp scene details.




Figure 4: Additional qualitative results on 7-Scenes Shotton et al. (2013) under 500 input views. The first column shows the input views, the middle column shows the reconstructed 3DGS, and the right column shows rendered novel views from LifeStreamGS.




Figure 5: Additional qualitative results on Tanks & Temples Knapitsch et al. (2017) under 500 input views. The first column shows the input views, the middle column shows the reconstructed 3DGS, and the right column shows rendered novel views from LifeStreamGS.

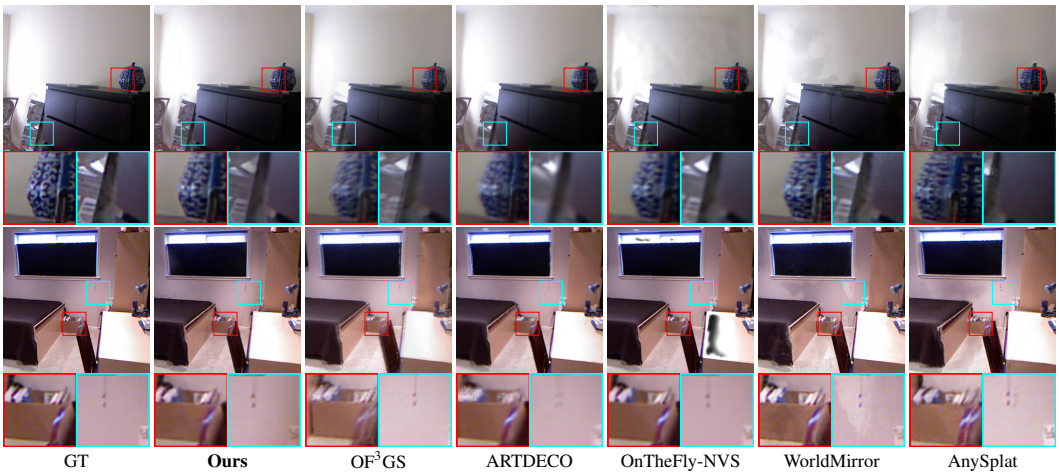


Figure 6: Qualitative zero-shot comparison on unseen NYUv2 Silberman et al. (2012) scenes. Our method produces clean novel view renderings and reduces visible artifacts under cross-dataset generalization.

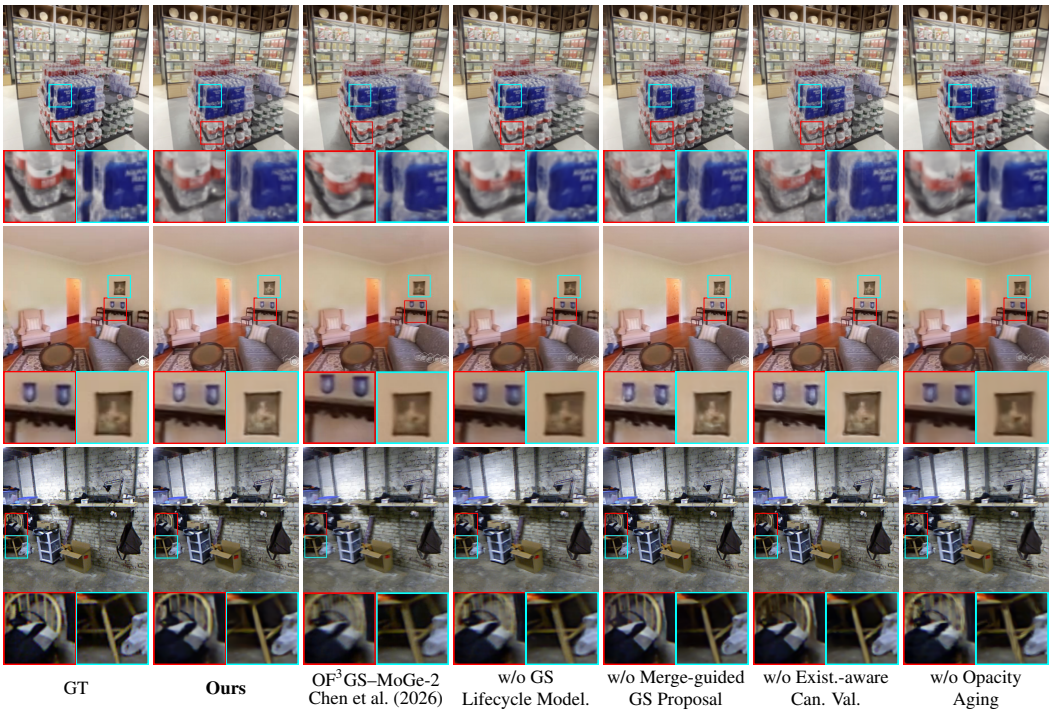


Figure 7: Qualitative ablation results under 5 input views on DL3DV-140 Ling et al. (2024) (top row), RE10K Zhou et al. (2018) (middle row), and NYUv2 Silberman et al. (2012) (bottom row) datasets.