PERSISTSPLAT: STREAMING FEED-FORWARD RECONSTRUCTION WITH PERSISTENT GAUSSIAN MEMORY

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent feed-forward 3D Gaussian Splatting (3DGS) strive to overcome the limitation of per-pixel Gaussian allocation to build more compact scene representations. However, most follow an offline paradigm that requires all views to be acquired in advance and jointly processed, thus unsuitable for streaming inputs. Existing online compression methods predict dense Gaussians for each frame and subsequently compress redundancy, which severely limits reconstruction quality and efficiency. We propose PersistSplat, a streaming feed-forward 3DGS architecture for constructing compact scene representations. Its core idea is to leverage incoming views continuously refine existing Gaussians, while restricting Gaussian generation to previously unobserved scene content; this allows Gaussian growth to be adaptively determined by scene scale rather than increasing with the view count. Specifically, PersistSplat maintains a Persistent Gaussian Memory for each Gaussian, with a fixed anchor and an updating historical observation set: for regions in the incoming view already covered by existing Gaussians, the model writes the current observations into the corresponding historical observation sets and uses the associated fixed anchors as queries to adaptively retrieve and aggregate historical context to refine existing Gaussians, avoiding redundant generation; for previously unobserved regions, it generates new Gaussians and initializes their anchors and historical observation sets, enabling incremental scene modeling. To ensure stable memory evolution, we further introduce Global Geometry Alignment and Global Feature Alignment to correct geometric drift and feature deviations. Experiments show that PersistSplat effectively reduces redundant Gaussians, allowing it to gradually converge to a scene-scale-dependent upper bound; while matching or outperforming existing offline and online baselines.

1 INTRODUCTION

Feed-forward 3D Gaussian Splatting (3DGS) has shown great potential for 3D reconstruction due to its efficient inference and strong generalization (Chen et al., 2024a; Zhang et al., 2024; Xu et al., 2025; 2026b). However, existing methods typically adopt per-pixel Gaussian allocation, tightly coupling the scene representation size with image resolution and view count. While preserving fine-grained details, this strategy often introduces substantial redundancy in smooth surfaces and overlapping regions, increasing computational and storage costs.

To build compact scene representations, recent works explore flexible Gaussian allocation mechanisms (Moreau et al., 2026; Xu et al., 2026a). TokenGS (Ren et al., 2026) regresses 3D Gaussians via learnable tokens, decoupling the Gaussian count from the number of pixels; ZPressor (Wang et al., 2025c) compresses multi-view features into a compact latent representation. Additionally, C3G (An et al., 2026) employs an epipolar Transformer to aggregate multi-view features and directly predict compact Gaussians without ground-truth poses; AdaptiveSplat (Singhal et al., 2026) adaptively allocates Gaussian density based on local texture information. However, constrained by the offline feed-forward paradigm, these methods all views to be acquired in advance and jointly processed, rendering them unsuitable for streaming inputs and limiting their practical scalability.

Some online compression methods seek to maintain compact scene representations during incremental modeling. LongSplat (Huang et al., 2026) fuses observations via a Gaussian-Image Representation and selectively prunes Gaussians for long-sequence reconstruction. StreamGS (Li et al.,

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.

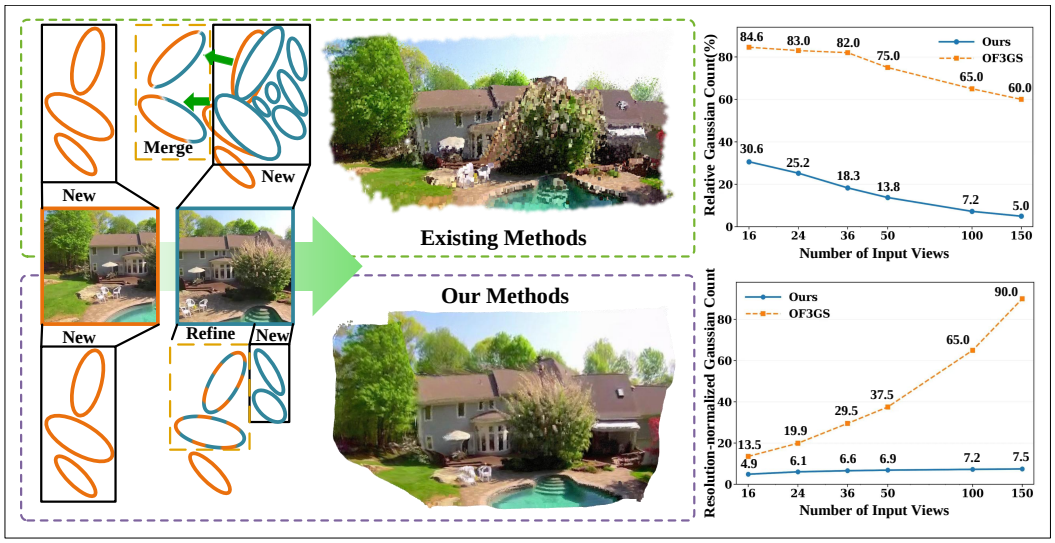


Figure 1: **Comparison of online reconstruction paradigms.** Existing online methods follow a paradigm of dense per-frame prediction followed by redundancy compression, causing the Gaussian count to increase with the number of input views; PersistSplat drives Gaussian generation by scene content, allowing Gaussian growth to be adaptively determined by the scene scale. As input views increase, the Relative Gaussian Count (ratio of Gaussians to total pixels across all views) continuously decreases, indicating a reduced proportion of new Gaussians from repeated observations and proving that Gaussian growth is progressively decoupled from the view count; meanwhile, the Resolution-normalized Gaussian Count (total Gaussians normalized by image resolution) stabilizes, showing that PersistSplat converges to a compact upper bound dictated by the scene scale.

2025) reduces redundancy through cross-frame Gaussian merging; OF3GS (Chen et al., 2026) recursively fuses current observations with historical representations, enabling Gaussian generation and incremental updates from pose-free image streams. However, these methods typically adopt a passive suppression strategy—densely predicting Gaussians for each frame and subsequently controlling the Gaussian count through cross-frame fusion or pruning. Such post-processing can induce geometric drift and scene degradation; more importantly, their Gaussian growth remains coupled with the view count, resulting in an increasing storage burden for long sequences.

To enable compact online scene reconstruction, we propose PersistSplat. As shown in Figure 1, our approach ensures Gaussian growth is no longer coupled with the input view count, but is adaptively determined by scene scale. Specifically, PersistSplat processes streaming inputs with a sliding window and maintains a Persistent Gaussian Memory for each Gaussian, consisting of a fixed anchor and an updating historical observation set: for regions in the incoming view already covered by existing Gaussians, it writes the current observations into the corresponding historical observation sets and uses the associated fixed anchors as queries to adaptively retrieve and aggregate historical context to refine the existing Gaussians, avoiding redundant generation; for previously unobserved regions, it generates new Gaussians and initializes their anchors and historical observation sets, enabling incremental scene modeling. To ensure geometric and feature consistency during memory evolution, we further introduce Global Geometry Alignment and Global Feature Alignment, which leverage cross-window constraints established by overlapping views to correct local geometric drift, and use fixed anchors as stable references to alleviate cross-window feature discrepancies. Experiments show that PersistSplat effectively reduces redundant Gaussians while maintaining compactness and achieving performance comparable to or better than existing baselines.

2 Related Work

2.1 Feed-forward 3D Gaussian Splatting

Feed-forward 3DGS has emerged as a mainstream paradigm for 3D reconstruction (Chen et al., 2024c; Jia et al., 2025; Miao et al., 2025). Early works such as pixelSplat (Charatan et al., 2024),

MVSplat (Chen et al., 2024b) and MonoSplat (Liu et al., 2025) addressed generalizable reconstruction from sparse views through cost volumes or epipolar geometric constraints, enabling high-quality novel view synthesis. More recently, methods including NoPoSplat (Ye et al., 2025), AnySplat (Jiang et al., 2025) and YoNoSplat (Ye et al., 2026) leverage pretrained backbones such as MASt3R (Leroy et al., 2024) and VGGT (Wang et al., 2025a) to extract cross-view geometric, enhancing generalization and robustness under wide-baseline or unconstrained settings. This readily introduces substantial redundancy in smooth or overlapping regions, resulting in significantly increased storage and computational overhead.

2.2 Compact 3D scene representations

Offline Feed-forward Method. These works fall into three main categories. The first generates a fixed number of Gaussian via global queries, using learnable tokens or global scene features, as in TokenGS (Ren et al., 2026), C3G (An et al., 2026) and GlobalSplat (Itkin et al., 2026). The second uses heuristic cues to guide Gaussian prediction, such as AdaptiveSplat (Singhal et al., 2026) EcoSplat (Park et al., 2026) and AnchorSplat (Zhang et al., 2026b). They adapt Gaussian density based on local texture complexity, 2D geometric primitives, or 3D anchor priors, balancing computational cost and reconstruction quality. The third introduces compression mechanisms to reduce scene representation scale, such as ZPressor (Wang et al., 2025c), TinySplat (Song et al., 2026) and Long-LRM (Chen et al., 2025). These methods filter redundant features via cross-view attention or information bottlenecks, compressing multi-view inputs into compact latent codes before decoding them into Gaussians. However, this paradigms assume complete scene observation and are thus unsuitable for streaming inputs, limiting their practicality.

Online Feed-forward Method Such methods maintain compact representations during incremental modeling (Wang et al., 2025d; Zhang et al., 2026a). LongSplat (Huang et al., 2026) uses Gaussian-Image Representation to fuse incoming observations and controls representation size through selective pruning; StreamGS (Li et al., 2025) and OF3GS (Chen et al., 2026) handle pose-free image streams via cross-frame Gaussian merging and online recursive fusion, respectively, reducing redundant Gaussian in overlapping regions. However, these methods are constrained by a passive suppression paradigm of "dense per-frame prediction followed by redundancy compression", where such post-processing operations as cross-frame fusion or pruning can readily induce geometric drift and scene degradation; moreover, since Gaussian growth remains coupled with the view count, computational efficiency for long sequences is difficult to guarantee. In contrast, PersistSplat constructs a Persistent Gaussian Memory that adaptively regulates Gaussian growth according to scene scale, reducing redundant Gaussian while enabling stable online reconstruction.

3 Method

3.1 Pipeline

Figure 2 shows the PersistSplat architecture. Given continuously incoming unposed views I, we employ a sliding window $\mathcal{W}$ for feed-forward processing. Specifically, a frozen feed-forward model $\mathcal{E}$ jointly encodes all views within $\mathcal{W}$ to predict corresponding local features, pointmaps, and camera poses: $\{\mathbf{F}^l, \mathbf{X}^l, \mathbf{P}^l\} = \mathcal{E}(\mathcal{W})$. Since each window is encoded independently, its local predictions inevitably exhibit geometric drift and feature deviations. To address this, we introduce Global Geometry Alignment (Sec. 3.2) and Global Feature Alignment (Sec. 3.3), which leverage cross-window geometric correspondences and stable reference features, respectively, to align current-window local predictions into a unified global space, yielding the global representations: $\{\mathbf{F}^g, \mathbf{X}^g, \mathbf{P}^g\}$.

To integrate streaming inputs, we maintain a Persistent Gaussian Memory (Sec. 3.4) for each Gaussian $\mathbf{g} \in \mathcal{G}$, comprising an anchor anchor $\mathbf{a_g}$ and an updating historical observation set $\mathcal{M}_\mathbf{g}$. Specifically, $\mathbf{a_g}$ is initialized with the corresponding global feature upon Gaussian creation and remains fixed, while $\mathcal{M}_\mathbf{g}$ accumulates observations across views. For each incoming view, we project existing Gaussians using its global pose $\mathbf{P}^g$ and determine the reliably covered region Ω^{cov} based on depth consistency between the projected Gaussians and the global pointmap $\mathbf{X}^g$. For each Gaussian whose projection falls within Ω^{cov}, the corresponding current observation is written into $\mathcal{M}_\mathbf{g}$, while $\mathbf{a_g}$ serves as a query to adaptively retrieve and aggregate relevant historical information for refining

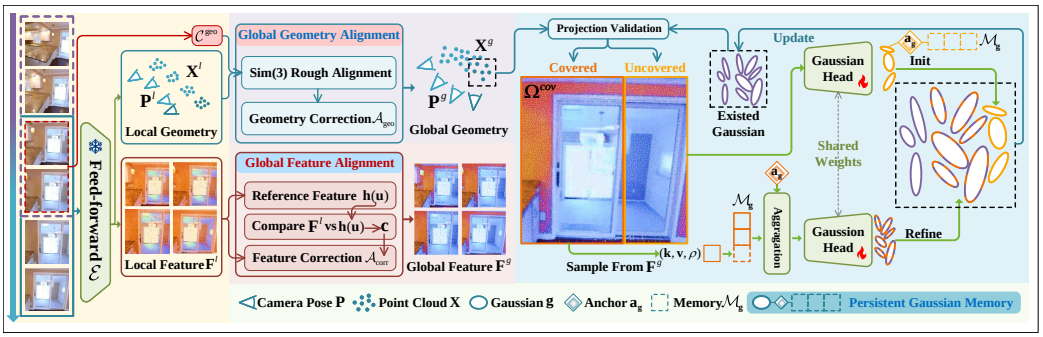


Figure 2: **Overview of PersistSplat.** Unposed image streams are processed in a sliding window $\mathcal{W}$ by a feed-forward model $\mathcal{E}$, yielding local features, pointmaps and poses $\{\mathbf{F}^l, \mathbf{X}^l, \mathbf{P}^l\}$. Global Geometry Alignment and Global Feature Alignment then correct them into unified global representations $\{\mathbf{F}^g, \mathbf{X}^g, \mathbf{P}^g\}$. To integrate streaming inputs, we maintain a Persistent Gaussian Memory for each Gaussian $\mathbf{g}$, consisting of a fixed anchor $\mathbf{a_g}$ and a historical observation set $\mathcal{M}_\mathbf{g}$: for the reliably covered region Ω^{cov}, current observation is sampled from $\mathbf{F}^g$, mapped to $\mathbf{k}$ and $\mathbf{v}$, and written into $\mathcal{M}_\mathbf{g}$ with its reliability ρ; the fixed anchor $\mathbf{a_g}$ serves as a query to adaptively retrieve and aggregate historical context for refining existing Gaussians; for novel scene content outside Ω^{cov}, new Gaussians are generated and their corresponding $\mathbf{a_g}$ and $\mathcal{M}_\mathbf{g}$ are initialized.

the Gaussian. Conversely, regions outside Ω^{cov} are treated as newly observed scene content, where new Gaussians are generated and their anchors and historical observation sets are initialized, enabling incremental scene modeling. Thus, PersistSplat continuously refines existing Gaussians with incoming views, restricting Gaussian generation to novel scene content; this allows Gaussian growth to be adaptively determined by scene scale, achieving compact and stable online reconstruction.

3.2 Global Geometry Alignment

Since different windows are independently encoded by the feed-forward network, their predicted geometries lie in distinct local coordinate systems; directly using these predictions to update the memory would severely compromise reconstruction consistency A straightforward solution is to estimate a $\mathrm{Sim}(3)$ transformation using overlapping views from adjacent windows to align the current window to the global coordinate system; however, the $\mathrm{Sim}(3)$ alignment only corrects global scale, rotation, and translation discrepancies, and cannot fully compensate for localized geometric distortions. To address this, we propose Global Geometry Alignment (GGA), which incorporates residual geometric correction upon the $\mathrm{Sim}(3)$ alignment to enhance cross-window geometric consistency. Specifically, we use the overlapping views to establish correspondences between the current local geometry and the global geometry from the previous window, and estimate a $\mathrm{Sim}(3)$ transformation accordingly. The coarse alignment of the pointmap and pose for each view is formulated as:

$$\widetilde{\mathbf{X}}, \widetilde{\mathbf{P}} = \mathcal{T}^{\text{sim}}\left(\mathbf{X}^l, \mathbf{P}^l; \mathcal{C}^{\text{geo}}\right) \tag{1}$$

where $\mathcal{C}^{\text{geo}}$ denotes the historical global geometry of the overlapping views. Subsequently, to further correct the remaining geometric errors, we introduce a geometry correction module $\mathcal{A}_{\text{geo}}$, which integrates the current local features, the coarsely aligned geometry and historical geometric context to produce a globally consistent geometry:

$$\mathbf{X}^g, \mathbf{P}^g = \mathcal{A}_{\text{geo}}\left(\mathbf{F}^l, \widetilde{\mathbf{X}}, \widetilde{\mathbf{P}}; \mathcal{C}^{\text{geo}}\right) \tag{2}$$

Notably, the global coordinate system is defined by the first window, while subsequent windows are recursively aligned to it.

3.3 Global Feature Alignment

Features independently encoded from different windows may exhibit deviations, causing the same spatial location to have inconsistent feature representations across observations; directly using such features to refine existing Gaussians would severely compromise the stability of memory updates.

As shown in Figure 3, Global Feature Alignment (GFA) aims to leverage the fixed anchors in the Persistent Gaussian Memory as stable references to correct the current local features into a globally consistent representation space. Specifically, for any pixel $\mathbf{u}$ within the covered region Ω^{cov} of incoming views, we aggregate the fixed anchors of existing Gaussians weighted by their rendering contributions to obtain the reference feature:

$$\mathbf{h}(\mathbf{u}) = \sum_{\mathbf{g}\in\mathcal{G}} r_{\mathbf{g}}(\mathbf{u})\mathbf{a}_{\mathbf{g}}, \qquad \text{where} \quad \mathbf{u} \in \Omega^{\text{cov}} \tag{3}$$

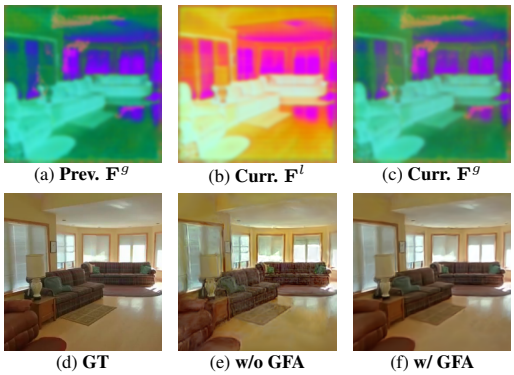
Here, $r_{\mathbf{g}}(\mathbf{u})$ denotes the normalized rendering contribution of Gaussian $\mathbf{g}$ to pixel $\mathbf{u}$. As the anchors remain fixed upon creation, $\mathbf{h}(\mathbf{u})$ serves as a stable reference. Next, we use a relation modeling module $\mathcal{A}_{\text{rel}}$ to compare the local feature with the reference feature and extract a view-level correction vector:

$$\mathbf{c} = \mathcal{A}_{\text{rel}}\left(\mathbf{F}^l(\mathbf{u}), \mathbf{h}(\mathbf{u})\right) \tag{4}$$

The correction module $\mathcal{A}_{\text{corr}}$ then then applies the broadcasted c to the entire image to yield the aligned global features:

$$\mathbf{F}^g = \mathbf{F}^l + \mathcal{A}_{\text{corr}}\left(\mathbf{F}^l, \text{Broadcast}(\mathbf{c})\right).$$

Notably, the local features of the initial window directly establish the initial representation space.



(a) **Prev.** $\mathbf{F}^g$ (b) **Curr.** $\mathbf{F}^l$ (c) **Curr.** $\mathbf{F}^g$

(d) **GT** (e) **w/o GFA** (f) **w/ GFA**

Figure 3: **Visualization of GFA and its effect.** For the same overlapping view, the first row shows the previously GFA-aligned global feature from the previous window, followed by its local feature and the GFA-aligned global feature in the current window. The second row compares the rendering results without and with GFA against the ground truth image.

3.4 Persistent Gaussian Memory

During online reconstruction, the same spatial location is observed repeatedly from different viewpoints. Independently generating new Gaussians for each observation would introduce substantial redundancy, while directly updating existing Gaussians using only the current observation overwrites the accumulated multi-view cues. To this end, we propose Persistent Gaussian Memory (PGM), which maintains a fixed anchor and an updating historical observation set for each Gaussian, and uses the fixed anchor as a query to adaptively retrieve and aggregate relevant historical context to continuously refine existing Gaussians.

Specifically, for a Gaussian $\mathbf{g}$, its anchor $\mathbf{a}_{\mathbf{g}}$ is initialized with the corresponding global feature upon creation and remains fixed; the historical observation set $\mathcal{M}_{\mathbf{g}}$ has a maximum capacity of J, is represented as: $\mathcal{M}_{\mathbf{g}} = \{\mathbf{k}_j, \mathbf{v}_j, \rho_j\}_{j=1}^{J}$, where $\mathbf{k}_j$ and $\mathbf{v}_j$ denote the feature representations stored for the j-th historical observation, and ρ_j denotes its reliability. For each incoming view, if the projection $\mathbf{u}_{\mathbf{g}}$ of an existing Gaussian $\mathbf{g}$ falls within Ω^{cov}, we sample the corresponding global feature as the current observation: $\mathbf{f}_{\mathbf{g}}^{\text{obs}} = \text{Sample}\left(\mathbf{F}^g, \mathbf{u}_{\mathbf{g}}\right)$, and linearly project it into the Key k and Value v to be written: $\mathbf{k} = W_K\mathbf{f}_{\mathbf{g}}^{\text{obs}}$, $\mathbf{v} = W_V\mathbf{f}_{\mathbf{g}}^{\text{obs}}$; meanwhile, the reliability ρ of the current observation is computed from the depth discrepancy between the projected depth of Gaussian $\mathbf{g}$ and the corresponding depth in the global pointmap $\mathbf{X}^g$. The resulting tuple $(\mathbf{k}, \mathbf{v}, \rho)$ is then written into the corresponding historical observation set $\mathcal{M}_{\mathbf{g}}$.

Since observations from different viewpoints vary in their relevance to the same Gaussian, direct average fusion may introduce unreliable information. Therefore, we use the fixed anchor as the query to adaptively retrieve and aggregate historical context from the updated historical observation set to refine the existing Gaussian. Given the query vector $\mathbf{q} = W_Q\mathbf{a}_{\mathbf{g}}$, the matching score for the j-th observation in the historical observation set is defined as:

$$e_j = \frac{\mathbf{q}^\top\mathbf{k}_j}{\sqrt{d}} + \lambda_1 \log(\rho_j + \epsilon) \tag{5}$$

where d denotes the feature dimension, and ϵ is a small constant for numerical stability. The historical observations are then weighted and aggregated according to their matching scores:

$$\mathbf{s} = \sum_{j=1}^{J} \alpha_j \mathbf{v}_j, \qquad \text{where} \quad \alpha_j = \frac{\exp(e_j)}{\sum_{j'=1}^{J}\exp(e_{j'})} \tag{6}$$

Next, we incorporate the aggregated historical context into the fixed anchor representation in a residual manner:

$$\bar{\mathbf{a}}_{\mathbf{g}} = \mathbf{a}_{\mathbf{g}} + \lambda_2 \tanh\left(W_S \mathbf{s}\right) \tag{7}$$

Here, $\bar{\mathbf{a}}_{\mathbf{g}}$ strictly serves as a conditional feature for the current Gaussian refinement, while the original anchor $\mathbf{a}_{\mathbf{g}}$ remains fixed. Finally, $\bar{a}_{\mathbf{g}}$ is fed into the Gaussian head to decode the updated Gaussian parameters, which directly replace the corresponding existing Gaussian:

$$\mathbf{g} \leftarrow \bar{\mathbf{g}}, \quad \text{where} \quad \bar{\mathbf{g}} = \mathcal{H}_{\mathrm{GS}}(\bar{a}_{\mathbf{g}}) \tag{8}$$

3.5 Training loss

The overall training objective is formulated as:

$$\mathcal{L} = \mathcal{L}_{\mathrm{render}} + \mathcal{L}_{\mathrm{GGA}} + \mathcal{L}_{\mathrm{GFA}} \tag{9}$$

where $\mathcal{L}_{\mathrm{render}}$ enforces consistency between the rendered results of the current Gaussians and the ground-truth images via MSE and LPIPS losses (Chen et al., 2024b). For $\mathcal{L}_{\mathrm{GGA}}$, we use the cross-window geometric correspondences established by overlapping views to constrain the GGA alignment results (Wang et al., 2024; 2025b):

$$\mathcal{L}_{\mathrm{GGA}} = \mathcal{L}_{\mathrm{plane}} + \lambda_{\mathrm{point}} \mathcal{L}_{\mathrm{point}} + \lambda_{\mathrm{pose}} \mathcal{L}_{\mathrm{pose}} \tag{10}$$

where $\mathcal{L}_{\mathrm{plane}}$ and $\mathcal{L}_{\mathrm{point}}$ respectively enforce local surface consistency and 3D positional consistency between the GGA-aligned pointmap and the historical global pointmap, while $\mathcal{L}_{\mathrm{pose}}$ constrains the consistency of the corresponding camera poses. For $\mathcal{L}_{\mathrm{GFA}}$, we constrain the GFA correction results by minimizing the cosine distance between the global features and the reference features within the reliably covered region Ω^{cov} (Gu et al., 2026) :

$$\mathcal{L}_{\mathrm{GFA}} = \frac{1}{|\Omega^{\mathrm{cov}}|} \sum_{\mathbf{u} \in \Omega^{\mathrm{cov}}} \left[1 - \cos\left(\mathbf{F}^g(\mathbf{u}), \mathbf{h}(\mathbf{u})\right)\right] \tag{11}$$

4 Experiments

4.1 Datasets and Baselines

We train on RE10K (Liu et al., 2021) and evaluate on RE10K and DL3DV-140 (Ling et al., 2024). Frozen AnySplat (Jiang et al., 2025) is used as feed-forward models to extract local geometry and features. Baselines include the offline compact feed-forward 3DGS methods EcoSplat (Park et al., 2026), C3G (An et al., 2026) and ZPressor (Wang et al., 2025c), as well as the online methods StreamGS (Li et al., 2025) and OF3GS (Chen et al., 2026).

4.2 Experiment Settings

We train the model using the AdamW optimizer with an initial learning rate of $1e^{-4}$ and a batch size of 1. The weights λ_{point}, λ_{pose}, λ_1, and λ_2 are set to 0.1, 0.05, 1.0, and 0.1. All comparisons are conducted at a spatial resolution of 224×224. The sliding-window size is set to 10 views, with 5 overlapping views between adjacent windows. All experiments are conducted on a single NVIDIA RTX A6000 GPU.

4.3 Comparisons with Baselines

Evaluation under Fixed Scene Scale We evaluate on RE10K under a fixed scene scale, with a fixed sequence length and varying inter-frame sampling rates. As shown in Figure 4 and Table 1, PersistSplat achieves rendering quality comparable to or better than AnySplat while requiring substantially fewer Gaussians, indicating that Persistent Gaussian Memory leverages multi-view observations to refine existing Gaussians and avoid redundant generation. Compared with offline baseline, C3G uses a fixed number of learnable Gaussian queries, which limits representation capacity in complex scenes. EcoSplat predicts dense Gaussians followed by importance-aware adaptive sparsification, achieving comparable performance to ours on short sequences. ZPressor leverages

Table 1: **Comparison of novel-view synthesis across input view counts.** Note that C3G only provides pretrained weights for 2K Gaussians.

Dataset	Method	Online	Input Views: 16				Input Views: 24				Input Views: 36			
			PSNR ↑	SSIM ↑	LPIPS ↓	#GS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	#GS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	#GS ↓
RE10K	AnySplat	×	23.69	0.795	0.191	562K	23.41	0.797	0.194	783K	23.37	0.781	0.197	903K
	C3G	×	20.78	0.662	0.268	2K	20.80	0.663	0.267	2K	20.13	0.642	0.275	2K
	EcoSplat	×	23.66	0.793	0.190	419K	23.48	0.788	0.195	550K	23.31	0.782	0.200	707K
	ZPressor	×	**23.89**	**0.846**	**0.151**	688K	**23.83**	**0.845**	**0.152**	688K	**23.87**	**0.845**	**0.152**	688K
	StreamGS	✓	16.23	0.532	0.381	681K	16.17	0.535	0.389	990K	16.62	0.539	0.387	1415K
	OF3GS	✓	18.69	0.580	0.395	1766K	18.45	0.572	0.413	2600K	18.29	0.568	0.423	3853K
	Ours	✓	23.75	0.805	0.193	321K	23.50	0.799	0.202	397K	23.41	0.790	0.196	423K
DL3DV	AnySplat	×	21.37	0.683	0.261	543K	20.65	0.647	0.298	782K	20.38	0.637	0.314	1034K
	C3G	×	18.65	0.483	0.381	2K	17.94	0.453	0.429	2K	17.61	0.442	0.451	2K
	EcoSplat	×	21.30	0.659	0.265	240K	20.56	0.617	0.298	361K	20.14	0.591	0.326	451K
	ZPressor	×	**22.59**	**0.744**	**0.163**	688K	**22.14**	**0.730**	**0.177**	688K	**21.74**	**0.715**	**0.188**	688K
	StreamGS	✓	17.51	0.529	0.385	642K	17.10	0.502	0.439	921K	16.84	0.485	0.473	1365K
	OF3GS	✓	18.71	0.550	0.358	2088K	18.14	0.532	0.397	3123K	17.64	0.503	0.424	4229K
	Ours	✓	21.92	0.714	0.251	260K	20.95	0.668	0.292	355K	20.54	0.646	0.312	439K

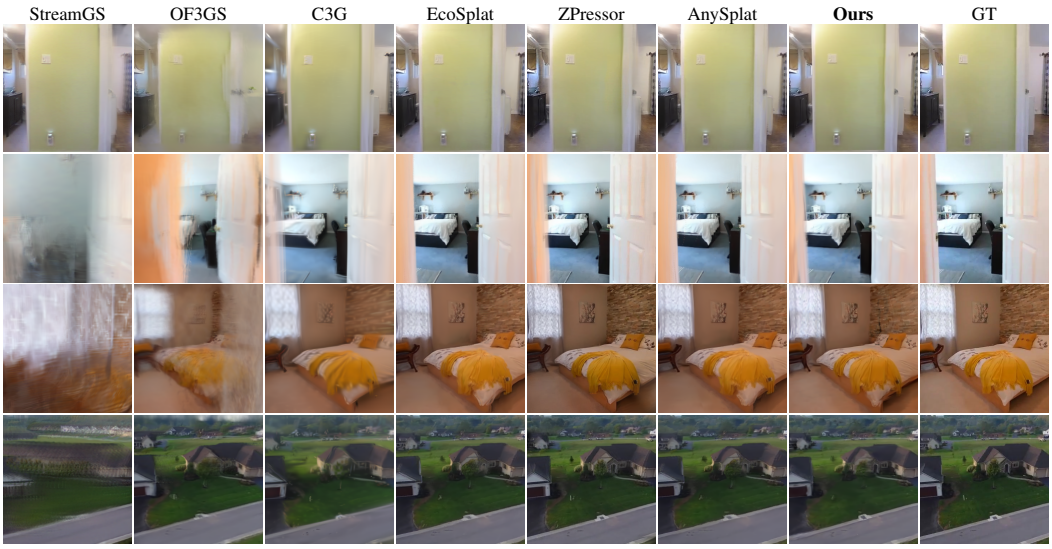


Figure 4: **Qualitative evaluation on RE10K under a fixed scene scale.**

ground-truth camera poses to compress multi-view features into a compact latent state, whereas PersistSplat achieves comparable performance without requiring pose inputs.

For online baselines, StreamGS estimates geometry with frozen DUSt3R and merges redundant Gaussians through cross-frame matching; its reliance on pairwise matching makes scale consistency difficult to maintain over long sequences. OF3GS adopts recursive Gaussian fusion with intrinsic recovery and point refinement to mitigate geometric errors, but struggles to persistently accumulate cross-view historical information, leading to performance degradation as the sequence grows. In contrast, PersistSplat continuously refines Gaussians using current observations and historical context, maintaining more stable reconstruction performance. Table 2 further shows PersistSplat's advantage under dense input streams: since Gaussian generation is driven by novel scene content, the Gaussian count gradually converges to an upper bound determined by scene scale as observations saturate; Gaussian growth in StreamGS and OF3GS remains coupled to the view count, failing to maintain a compact representation over long sequences.

Evaluation under Expanding Scene Scale We evaluate on DL3DV under an expanding scene scale, with a fixed inter-frame interval and increasing sequence length. As shown in Figure 5, Table 1 and Table 2, while AnySplat jointly encodes all views to model dense cross-view correspondences, whereas PersistSplat performs online reconstruction using only current observations and historical context, yet achieves comparable rendering quality with fewer Gaussians. When EcoSplat is adjusted to use a Gaussian count comparable to ours, PersistSplat demonstrates superior performance. Moreover, as the scene scale continues to expand, PersistSplat maintains relatively stable perfor-

Table 2: **Comparison of novel-view synthesis on long-sequence inputs.**

Dataset	Method	Input Views: 50				Input Views: 100				Input Views: 150			
		PSNR ↑	SSIM ↑	LPIPS ↓	#GS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	#GS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	#GS ↓
RE10K	StreamGS	15.91	0.498	0.400	1802K	15.68	0.489	0.410	4259K	15.53	0.419	0.418	5210K
	OF3GS	19.46	0.573	0.359	4895K	19.35	0.567	0.364	8484K	19.22	0.542	0.383	11748K
	Ours	**23.22**	**0.784**	**0.204**	**451K**	**21.90**	**0.743**	**0.229**	**475K**	**20.87**	**0.705**	**0.259**	**489K**
DL3DV	StreamGS	16.30	0.448	0.507	1966K	15.04	0.409	0.562	4587K	14.91	0.412	0.572	6881K
	OF3GS	16.77	0.493	0.486	5547K	15.96	0.428	0.502	10834K	15.11	0.408	0.533	15664K
	Ours	**18.22**	**0.519**	**0.445**	**551K**	**17.55**	**0.457**	**0.493**	**919K**	**17.02**	**0.430**	**0.508**	**1072K**

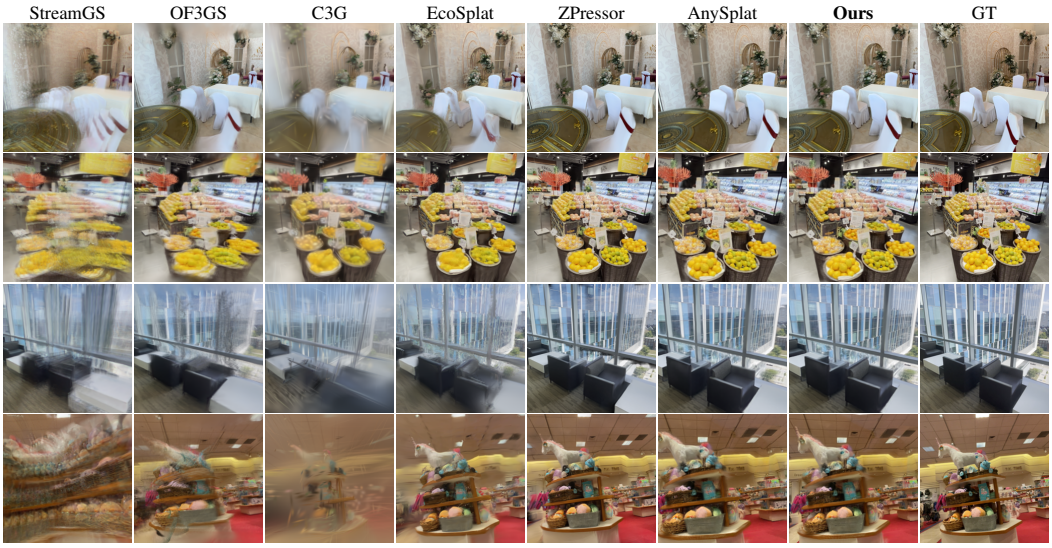


Figure 5: **Qualitative evaluation on DL3DV under an expanding scene scale.**

mance and consistently outperforms StreamGS and OF3GS with a more compact scene representation, demonstrating its robustness and scalability.

4.4 Ablation Study

Effect of Global Alignment As shown in Table 3 and Figure 7, removing Global Geometry Alignment (GGA) and relying solely on the $\mathrm{Sim}(3)$ transformation cannot fully compensate for geometric misalignment, whereas introducing the geometric correction module significantly improves cross-window alignment accuracy. As shown in Figure 6, without Global Feature Alignment (GFA), encoding each window independently introduces feature deviations that hinder the continuous refinement of Gaussians. By utilizing fixed anchors as stable references for feature correction effectively mitigates cross-window feature inconsistencies.

Table 3: **Ablation on GGA and GFA.**

	GGA	GFA	PSNR ↑	SSIM ↑	LPIPS ↓	#GS ↓
1	×	×	21.44	0.735	0.283	462K
2	✓	×	21.68	0.743	0.269	451K
3	×	✓	22.06	0.752	0.230	462K
4	✓	✓	**23.22**	**0.784**	**0.204**	**451K**

Effect of Persistent Gaussian Memory As shown in Figure 8 and Table 4, removing Persistent Gaussian Memory (PGM) and directly predicting and accumulating Gaussians frame by frame (Dir. Accum.) leads to substantial redundancy. Further applying opacity-based pruning (Opac. Prun.) may discard scene information, while voxel-based Gaussian merging (Voxel Merge.) can overwrite previously accumulated multi-view cues; both lead to noticeable performance degradation. After introducing PGM, directly fusing the multi-view cues stored in $\mathcal{M}_{\mathbf{g}}$ for refinement (Mem. Fus.) may introduce noise or unreliable observations. In contrast, the complete

Table 4: **Ablation on PGM.** Details of each variant are in the Appendix.

	Variant	PSNR ↑	SSIM ↑	LPIPS ↓	#GS ↓
1	Dir. Accum.	22.67	0.776	0.211	2508K
2	Opac. Prun.	21.45	0.731	0.280	451K
3	Voxel Merge.	22.09	0.745	0.238	451K
4	Mem. Fus.	22.45	0.762	0.212	451K
5	**Full PGM**	**23.22**	**0.784**	**0.204**	**451K**



Figure 6: **Qualitative ablation of global alignment components.** "Baseline" denotes the setting without GGA and GFA.

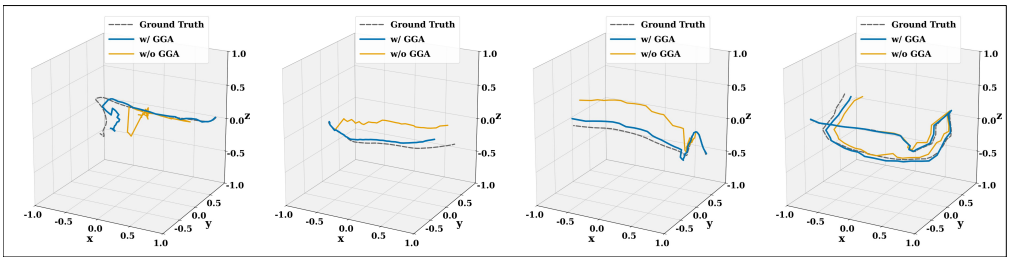


Figure 7: **Camera trajectory visualization for Global Geometry Alignment ablation.**



Figure 8: **Qualitative ablation of Persistent Gaussian Memory.**

PGM uses fixed anchors as queries to adaptively retrieve and aggregate relevant historical context, thereby maintaining more consistent geometric and appearance representations.

5 Conclusion and Limitations

Conclusion: To enable compact online scene reconstruction, we propose PersistSplat, where Gaussian growth is driven by scene content rather than the view count. Persistent Gaussian Memory (PGM) continuously refines existing Gaussians using fixed anchors and historical observations to avoid redundant generation, while Global Geometry Alignment (GGA) and Global Feature Alignment (GFA) correct geometric drift and feature deviations. **Limitation:** PGM's historical observation set is dynamically updated with streaming input, but removed observations may contain complementary cues beneficial to future views, potentially affecting subsequent refinement. PersistSplat must balance cross-window alignment accuracy and inference efficiency: too few overlapping frames weaken geometric alignment, while too many reduce inference efficiency. Future work will explore flexible historical observation update mechanisms and adaptive window-overlap strategies to better balance long-term information utilization and inference efficiency.

AI USE STATEMENT

In this study, generative AI tools were strictly utilized as assistive utilities to improve the efficiency of partial code implementation and to polish the language of the manuscript. All AI-assisted code and text were rigorously tested, reviewed, and edited by the authors. The core conceptualization, methodological design, experimental execution, and scientific conclusions were independently led and completed by the authors, who assume full responsibility for the entirety of this paper.

REPRODUCIBILITY STATEMENT

To ensure the reproducibility of our work, we provide detailed descriptions of the method design, training settings, and experimental configurations in the main paper, along with the core implementation code in the supplementary material. We will further organize and release the relevant code and configurations in the final version to facilitate result verification and future research.

ETHICS STATEMENT

This research primarily focuses on advancing 3D reconstruction methods. It does not involve sensitive personal information or private data, nor does it present apparent risks related to safety, bias, or potentially harmful applications. All datasets used in this work are publicly available for research purposes, and we comply with their corresponding licenses and usage guidelines.

REFERENCES

Honggyu An, Jaewoo Jung, Mungyeom Kim, Chaehyun Kim, Minkyeong Jeon, Jisang Han, Kazumi Fukuda, Takuya Narihira, Hyunah Ko, Junsu Kim, Sunghwan Hong, Yuki Mitsufuji, and Seungryong Kim. Learning Compact 3D Representations from Feed-Forward Novel View Synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 363–373, June 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 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19457–19467, 2024.

Anpei Chen, Haofei Xu, Stefano Esposito, Siyu Tang, and Andreas Geiger. LaRa: Efficient Large-Baseline Radiance Fields. In *European Conference on Computer Vision*, pp. 338–355. Springer, 2024a. doi: 10.1007/978-3-031-73247-8_20.

Ruiyang Chen, Feiran Li, Chu Zhou, Zonglin Li, Zhanyu Ma, and Heng Guo. OF^{3}GS: On-the-Fly Feed-Forward 3D Gaussian Splatting from Unposed Images, 2026. URL `https://arxiv.org/abs/2606.03254`.

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, 2024b. doi: 10.1007/978-3-031-72664-4_21.

Yuedong Chen, Chuanxia Zheng, Haofei Xu, Bohan Zhuang, Andrea Vedaldi, Tat-Jen Cham, and Jianfei Cai. MVSplat360: Feed-Forward 360 Scene Synthesis from Sparse Views. In *Advances in Neural Information Processing Systems*, volume 37, pp. 107064–107086. Curran Associates, Inc., 2024c. doi: 10.52202/079017-3399.

Ziwen Chen, Hao Tan, Kai Zhang, Sai Bi, Fujun Luan, Yicong Hong, Fuxin Li, and Zexiang Xu. Long-LRM: Long-sequence Large Reconstruction Model for Wide-coverage Gaussian Splats. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4349–4359, 2025.

Meiying Gu, Jiawei Zhang, Jiahe Li, Xiaohan Yu, Haonan Luo, Jin Zheng, and Xiao Bai. SparseSurf: Sparse-View 3D Gaussian Splatting for Surface Reconstruction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(6):4311–4319, 2026. doi: 10.1609/aaai.v40i6.42428.

Guichen Huang, Ruoyu Wang, Xiangjun Gao, Che Sun, Yuwei Wu, Shenghua Gao, and Yunde Jia. LongSplat: Online Generalizable 3D Gaussian Splatting from Long Sequence Images. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(6):4994–5002, 2026. doi: 10.1609/aaai.v40i6.42504.

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.

Heng Jia, Linchao Zhu, and Na Zhao. H3R: Hybrid Multi-view Correspondence for Generalizable 3D Reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7655–7665, 2025.

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*, 44(6):1–16, 2025. doi: 10.1145/3763326.

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. doi: 10.1007/978-3-031-73220-1_5.

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. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 25841–25850, 2025.

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 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22160–22169, 2024.

Andrew Liu, Richard Tucker, Varun Jampani, Ameesh Makadia, Noah Snavely, and Angjoo Kanazawa. Infinite Nature: Perpetual View Generation of Natural Scenes From a Single Image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 14458–14467, 2021.

Yifan Liu, Keyu Fan, Weihao Yu, Chenxin Li, Hao Lu, and Yixuan Yuan. MonoSplat: Generalizable 3D Gaussian Splatting from Monocular Depth Foundation Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21570–21579, 2025.

Sheng Miao, Jiaxin Huang, Dongfeng Bai, Xu Yan, Hongyu Zhou, Yue Wang, Bingbing Liu, Andreas Geiger, and Yiyi Liao. EVolSplat: Efficient Volume-based Gaussian Splatting for Urban View Synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11286–11296, 2025.

Arthur Moreau, Richard Shaw, Michal Nazarczuk, Jisu Shin, Thomas Tanay, Zhensong Zhang, Songcen Xu, and Eduardo Pérez-Pellitero. Off The Grid: Detection of Primitives for Feed-Forward 3D Gaussian Splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11756–11766, 2026.

Jongmin Park, Minh-Quan Viet Bui, Juan Luis Gonzalez, Jaeho Moon, Jihyong Oh, and Munchurl Kim. EcoSplat: Efficiency-controllable Feed-forward 3D Gaussian Splatting from Multi-view Images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26010–26020, 2026.

Jiawei Ren, Michal Jan Tyszkiewicz, Jiahui Huang, and Zan Gojcic. TokenGS: Decoupling 3D Gaussian Prediction from Pixels with Learnable Tokens. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15365–15375, June 2026.

Badrinath Singhal, K. G. Srihari, Sreehari Iyer, Ankit Dhiman, and Venkatesh Babu Radhakrishnan. AdaptiveSplat: Texture Aware Controllable 3D Gaussian Allocation for Feed-Forward Reconstruction. In *European Conference on Computer Vision*, pp. 503–522. Springer, 2026. doi: 10.1007/978-3-032-37281-9_31.

Zetian Song, Jiaye Fu, Jiaqi Zhang, Xiaohan Lu, Chuanmin Jia, Siwei Ma, and Wen Gao. TinySplat: Feedforward Approach for Generating Compact 3D Scene Representation. *IEEE Transactions on Circuits and Systems for Video Technology*, 36(4):5567–5580, 2026. doi: 10.1109/tcsvt.2026.3664794.

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, 2025a.

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, 2025b.

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, 2024.

Weijie Wang, Donny Y Chen, Zeyu Zhang, Duochao Shi, Akide Liu, and Bohan Zhuang. ZPressor: Bottleneck-Aware Compression for Scalable Feed-Forward 3DGS. In *Advances in Neural Information Processing Systems*, volume 38, pp. 113407–113436. Curran Associates, Inc., 2025c. doi: 10.52202/085713-3782.

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 *Advances in Neural Information Processing Systems*, volume 38, pp. 76311–76341. Curran Associates, Inc., 2025d. doi: 10.52202/085713-2563.

Haofei Xu, Songyou Peng, Fangjinhua Wang, Hermann Blum, Daniel Barath, Andreas Geiger, and Marc Pollefeys. DepthSplat: Connecting Gaussian Splatting and Depth. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16453–16463, 2025.

Haofei Xu, Daniel Barath, Andreas Geiger, and Marc Pollefeys. ReSplat: Learning Recurrent Gaussian Splatting. In *European Conference on Computer Vision*, pp. 364–382. Springer, 2026a. doi: 10.1007/978-3-032-37435-6_20.

Muyu Xu, Fangneng Zhan, Xiaoqin Zhang, Ling Shao, and Shijian Lu. MuSASplat: Efficient Sparse-View 3D Gaussian Splats via Lightweight Multi-Scale Adaptation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(14):11343–11351, 2026b. doi: 10.1609/aaai.v40i14.38115.

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. In *International Conference on Learning Representations*, volume 2025, pp. 54009–54033, 2025.

Botao Ye, Boqi Chen, Haofei Xu, Daniel Barath, and Marc Pollefeys. YoNoSplat: You Only Need One Model for Feedforward 3D Gaussian Splatting. In *International Conference on Learning Representations*, volume 2026, pp. 39852–39871, 2026.

Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. GS-LRM: Large Reconstruction Model for 3D Gaussian Splatting. In *European Conference on Computer Vision*, pp. 1–19. Springer, 2024. doi: 10.1007/978-3-031-72670-5_1.

Renhe Zhang, Yuyang Tan, Jingyu Gong, Zhizhong Zhang, Lizhuang Ma, Yuan Xie, and Xin Tan. S2GS: Streaming Semantic Gaussian Splatting for Online Scene Understanding and Reconstruction. *arXiv preprint arXiv:2603.14232*, 2026a.

Xiaoxue Zhang, Xiaoxu Zheng, Yixuan Yin, Tiao Zhao, Kaihua Tang, Michael Bi Mi, Zhan Xu, and Dave Zhenyu Chen. AnchorSplat: Feed-Forward 3D Gaussian Splatting With 3D Geometric Priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18924–18933, June 2026b.

A Appendix

A.1 Reliable Coverage Identification

To determine the reliably covered region Ω^{cov} in the incoming view, we project existing Gaussians using the global pose $\mathbf{P}^g$ and determine their validity based on depth consistency with the global pointmap $\mathbf{X}^g$. Specifically, let $\mathbf{u_g}$ denote the projected pixel location of an existing Gaussian $\mathbf{g}$, with its center depth $d_{\mathbf{g}}$ and its depth extent along the ray $\delta_{\mathbf{g}}^{\text{gs}}$, derived from the Gaussian covariance. Correspondingly, let $d_{\mathbf{g}}^{\text{obs}}$ and $\delta_{\mathbf{g}}^{\text{obs}}$ denote the observed depth and its tolerance of the global pointmap $\mathbf{X}^g$ at $\mathbf{u_g}$. A reliable geometric correspondence is established when the two depth ranges overlap:

$$\chi_{\mathbf{g}}^{\text{cov}} = \mathbb{I}\left[\left|d_{\mathbf{g}} - d_{\mathbf{g}}^{\text{obs}}\right| \le \delta_{\mathbf{g}}^{\text{gs}} + \delta_{\mathbf{g}}^{\text{obs}}\right] \tag{12}$$

where $\mathbb{I}[\cdot]$ is the indicator function. The set of projected locations with such reliable correspondences constitutes the reliably covered region in the current view:

$$\Omega^{\text{cov}} = \left\{\mathbf{u_g} \mid \chi_{\mathbf{g}}^{\text{cov}} = 1\right\} \tag{13}$$

A.2 Historical Observation Update

Since the historical observation set $\mathcal{M}_{\mathbf{g}}$ of each Gaussian has a fixed capacity $J = 8$, we dynamically maintain it to continuously accumulate observations from different viewpoints. In addition to the key $\mathbf{k}$, value $\mathbf{v}$, and reliability ρ, each historical observation records the corresponding viewing direction $\mathbf{d}$ and log-depth ℓ:

$$\mathcal{M}_{\mathbf{g}} = \{\mathbf{k}_j, \mathbf{v}_j, \mathbf{d}_j, \ell_j, \rho_j\}_{j=1}^{J} \tag{14}$$

Here, the viewing direction $\mathbf{d}_j$ is computed from the relative position between the camera center of the corresponding view and the Gaussian center, while the log-depth ℓ_j is derived from the projected depth of the Gaussian in that view. Both are written into $\mathcal{M}_{\mathbf{g}}$ alongside $\mathbf{k}_j$, $\mathbf{v}_j$, and ρ_j. For an incoming observation $(\mathbf{k}, \mathbf{v}, \mathbf{d}, \ell, \rho)$, if there exists a historical slot j satisfying:

$$\cos(\mathbf{d}, \mathbf{d}_j) \ge \tau_{\text{dir}}, \qquad |\ell - \ell_j| \le \tau_{\text{depth}} \tag{15}$$

where τ_{dir} and τ_{depth} are empirically set to 0.97 and 0.1, respectively. The incoming observation is regarded as consistent with the historical observation in both viewing direction and depth, and the corresponding slot is updated according to its reliability ρ; otherwise, it is stored in an empty slot as a new historical observation. When $\mathcal{M}_{\mathbf{g}}$ is full, we evaluate the viewpoint similarity among historical observations based on the viewing direction $\mathbf{d}_j$. We replace the one with lower reliability ρ_j among the historical observations with the most similar viewing directions, and then write the incoming observation into the corresponding slot. Consequently, $\mathcal{M}_{\mathbf{g}}$ can retain complementary multi-view observations under a fixed capacity J, preventing historical information from being simply overwritten by recent observations.

A.3 Persistent Gaussian Memory Ablation Settings

This section describes the implementation details of the Persistent Gaussian Memory (PGM) ablation variants presented in Sec. 4.4. Specifically, while keeping the Global Geometry Alignment and Global Feature Alignment components intact, we construct the following variants:

Direct Accumulation (Dir. Accum.) This variant completely removes PGM. For each incoming view, the model directly predicts the corresponding Gaussians and accumulates them into the global Gaussian set, without applying any compression or merging operations.

Opacity Pruning (Opac. Prun.) This setting performs opacity-based pruning on the Gaussian set obtained by Direct Accumulation. We rank all Gaussians according to their opacity and remove the lowest 82%, ultimately retaining approximately 451K Gaussians to match the representation scale of the full model. Notably, this operation is executed only once after the entire sequence has been processed, rather than dynamically during the streaming input.

Voxel Merging (Voxel Merge.) This variant replaces the PGM with a nearest-neighbor merging operation in 3D space. We discretize the 3D space into fixed-size voxels; for multiple Gaussians



Figure 9: **Qualitative ablation of Fixed Anchor.**

located within the same voxel, the model performs an opacity-weighted average to explicitly merge their Gaussian parameters.

Memory Fusion (Mem. Fus.) This variant retains the historical observation set in PGM but removes the adaptive retrieval and aggregation based on the fixed anchor. For the multi-view cues stored in the historical observation set $\mathcal{M}_g$, we directly perform equal-weight fusion and use it to refine the Gaussian.

A.4 Efficiency Analysis

As shown in Table 5, as the sequence length increases, frequent cross-window alignment and continuous Gaussian refinement incur additional computational overhead, leading to a decline in inference speed; however, this decline gradually levels off as scene observations saturate. Meanwhile, memory usage remains relatively stable and does not grow linearly with sequence length, demonstrating the scalability of PersistSplat.

Table 5: **Computational efficiency under different sequence lengths.**

Views	Wins	FPS ↑	Memory ↓
20	3	15.12	11.69GB
50	9	9.38	12.21GB
100	19	9.07	12.62GB
150	29	8.95	13.03GB

A.5 Effect of Fixed Anchor

As shown in Figure 9 and Table 6, allowing anchors $\mathbf{a_g}$ to update continuously with observations significantly degrades reconstruction performance. Without anchors serving as stable references, Global Feature Alignment fails to effectively correct cross-window feature deviations. Concurrently, the continuous drift of query priors prevents PGM from stably matching multi-view cues, rendering the aggregated historical context unreliable.

Table 6: **Ablation on fixed anchor.**

Fixed Anchor	PSNR ↑	SSIM ↑	LPIPS ↓	#GS ↓
×	19.45	0.657	0.310	451K
✓	**23.22**	**0.784**	**0.204**	**451K**