

I-GaSp3R: Incremental Gaussian Splatting for 3D Reconstruction

Tobias Steinle
Technical University of Munich
Munich, Germany
tobias.steinle@tum.de

Osama Salem
Technical University of Munich
Munich, Germany
osama.salem@tum.de

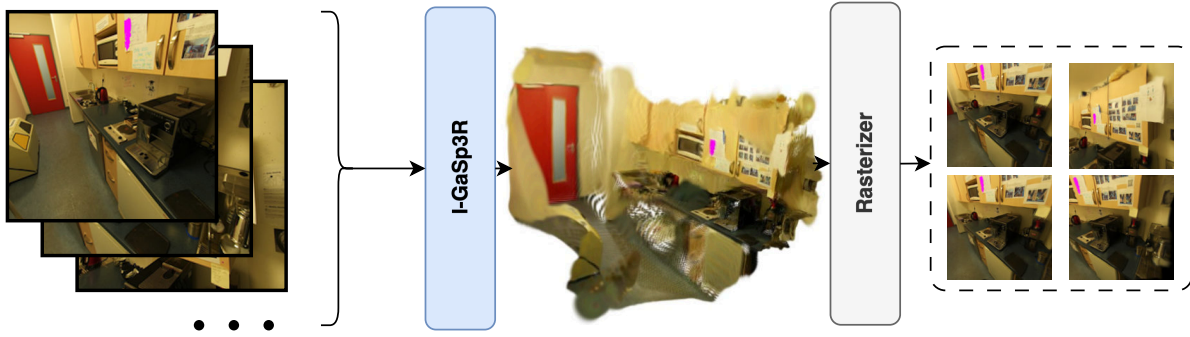


Figure 1. **Overview of I-GaSp3R.** Given a sequence of unposed images of variable length, I-GaSp3R incrementally reconstructs a set of 3D Gaussian primitives in a single feed-forward pass without any pre-processing or intermediate pose estimation. I-GaSp3R operates at 21 FPS on a single RTX 3080 GPU, enabling real-time novel-view synthesis from numerous input images through differentiable rasterization.

Abstract

Recent progress in 3D scene reconstruction and novel-view synthesis has led to significant improvements in rendering quality. However, many state-of-the-art methods depend on explicit camera calibration and offline optimization, which restricts their application in dynamic environments. We propose Incremental Gaussian Splatting for 3D Reconstruction (I-GaSp3R), a novel framework that enables incremental, real-time, and pose-free 3D scene reconstruction alongside novel-view synthesis. By leveraging an efficient 3D Gaussian representation combined with a spatial memory network, I-GaSp3R achieves superior rendering quality and computational efficiency. Extensive evaluations on the ScanNet++ dataset demonstrate that our approach outperforms existing methods, marking a significant step toward practical real-time 3D scene understanding.

1. Introduction

Gaussian Splatting has recently emerged as an efficient alternative to volumetric approaches for real-time 3D scene reconstruction and rendering. Unlike NeRF, which relies on computationally expensive volumetric ray marching, Gaussian Splatting represents scenes using 3D Gaussian primitives, allowing for fast rasterization. Despite these advan-

tages, most existing methods assume known camera poses and require offline optimization, limiting their applicability in dynamic or unstructured settings.

We introduce **Incremental Gaussian Splatting for 3D Reconstruction (I-GaSp3R)**, a novel approach that enables real-time, pose-free scene reconstruction and novel-view synthesis. Our framework integrates the spatial memory module from Spann3R with the Gaussian representation from Splatt3R, enabling a feed-forward architecture that runs at 21 FPS on an RTX 3080. To ensure stable, high-quality reconstructions, we propose:

- **Real-time Feed-forward 3D Gaussian Splatting:** An efficient pipeline that incrementally reconstructs 3D scenes without iterative optimization.
- **Temporal Consistency:** A novel loss function that mitigates aliasing and ensures stability across frames.
- **Improved Training Stability:** Enhanced robustness via masking techniques (frustum overlap & depth matching) and tailored regularization strategies.

2. Related Work

Traditional Scene Rendering and NeRF Early approaches to novel-view synthesis relied on light fields, both dense [12, 15] and unstructured [1], which captured the ap-

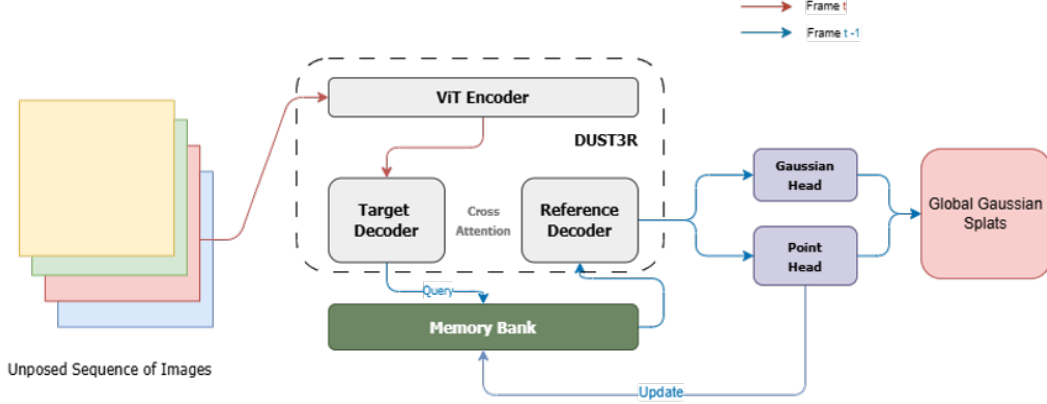


Figure 2. **Architecture Overview.** Each frame is encoded by a ViT, and the extracted features interact with a memory bank to maintain spatial consistency. The heads then predict per-frame Gaussian attributes, which accumulated for real-time rendering.

pearance of scenes from multiple viewpoints. Structure-from-Motion (SfM) [19] enabled 3D reconstruction from photo collections by estimating sparse point clouds, while Multi-View Stereo (MVS) techniques advanced dense reconstruction capabilities. These methods paved the way for geometry-based view synthesis approaches [3, 10], which re-project and blend images based on estimated scene geometry. However, such methods struggle with artifacts in regions of missing or inaccurate geometry.

Neural rendering has mitigated these limitations by learning data-driven representations that improve rendering quality and robustness. Early deep learning-based view synthesis methods [11] still depended on MVS pipelines and suffered from temporal instability. Volumetric methods using differentiable density fields introduced ray-marching techniques, which, while effective, were computationally expensive. Neural Radiance Fields (NeRF) [16] significantly improved photorealism through volumetric rendering with importance sampling and positional encoding, though querying such models remains computationally intensive for real-time applications.

Feed-forward Novel-view Synthesis Recent efforts focus on feed-forward architectures for novel-view synthesis, reducing reliance on expensive optimization procedures. Gaussian Splatting approaches predict 3D scene representations directly from 2D images. Splatter Image generates pixel-aligned Gaussians from single images, while pixel-Splat [2] extends this to image pairs, though it struggles with maintaining accurate geometry in noisy conditions. MVSplat [4] handles variable-length sequences but requires known camera poses, limiting its flexibility in dynamic environments.

Feed-forward 3D structure recovery also includes depth estimation [6, 9], optical flow [20], and point tracking [7, 13]. DUST3R [22] predicts 3D point maps from image pairs

but requires global alignment for long sequences, while MAST3R [14] improves feature matching for 3D tasks. Spann3R [21] introduces a spatial memory mechanism that eliminates the need for global alignment, enabling incremental 3D reconstruction.

Our method, **I-GaSp3R**, builds upon Spann3R by incorporating the Gaussian head from Splatt3R to predict rich 3D Gaussian attributes. This integration results in an incremental, pose-free, feed-forward, and online pipeline for real-time 3D reconstruction and novel-view synthesis, with potential applications in augmented reality, virtual reality, and robotics.

3. Method

In this section, we introduce I-GaSp3R, a *pose-free, feed-forward* framework for real-time 3D scene reconstruction and novel-view synthesis. Our method incrementally processes an unposed image sequence $\{I_t\}_{t=1}^T$, using a spatial memory mechanism to integrate new frames without requiring global optimization. The scene is represented as a growing set of 3D Gaussian splats, which are rendered via differentiable rasterization.

3.1. Architecture

ViT-based Encoder and Spatial Memory Each incoming frame I_t is passed through a Vision Transformer [17] to obtain latent features f_t . A *spatial memory* network [21] fuses these features with context from prior frames, allowing incremental reconstruction without explicit pose information. Specifically:

- The *Target Decoder* produces a query feature f_t^q that retrieves relevant spatial information from the memory.
- The *Reference Decoder* refines features from the previous frame, ensuring smooth scene updates.
- A *Memory Bank* stores and updates past information using attention-based sparsification.

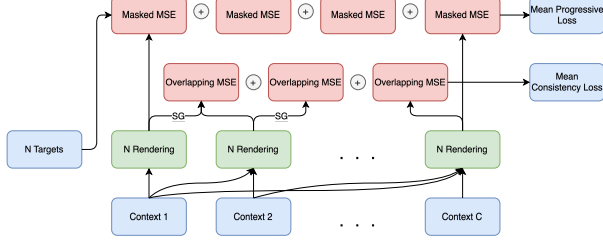


Figure 3. **Training Framework.** For each frame, the accumulated Gaussians are used to render an image $\hat{I}_i$, which is compared to the target I_i via a masked photometric loss. Additionally, overlapping regions between consecutive renderings are penalized for temporal consistency.

This enables a *causal* reconstruction process, where earlier frames establish scene structure and later frames refine details.

Dual Prediction Heads

- **Point Head:** Predicts 3D Gaussian *means* (i.e., per-pixel pointmaps) and a *confidence score* that determines how strongly each prediction contributes to the global scene.
- **Gaussian Head:** Predicts *opacity*, *scale*, *rotation*, and *color* for each splat, enabling efficient differentiable rendering.

These predictions define a set of 3D Gaussian primitives that are accumulated in a global representation for rendering novel views.

3.2. Progressive Training with Temporal Consistency

Incremental Rendering and Supervision For each context frame I_i , we render a predicted image $\hat{I}_i$ using the accumulated set of Gaussians. We enforce supervision using two complementary objectives:

- A **masked photometric loss** ensures fidelity to the ground-truth target while ignoring unobserved regions:

$$\mathcal{L}_{\text{photo}} = \sum_{i=1}^C \mathcal{L}_{\text{MSE}}(M \cdot \hat{I}_i, M \cdot I_i) + \mathcal{L}_{\text{LPIPS}}(M \cdot \hat{I}_i, M \cdot I_i). \quad (1)$$

where M is a visibility mask as explained in Section 6.

- A **temporal consistency loss** (Fig. 3) enforces smooth transitions between consecutive renderings:

$$\mathcal{L}_{\text{temp}} = \sum_{i=1}^{C-1} \mathcal{L}_{\text{MSE}}(\hat{I}_i, \hat{I}_{i+1}) + \mathcal{L}_{\text{LPIPS}}(\hat{I}_i, \hat{I}_{i+1}). \quad (2)$$

These losses ensure that earlier frames establish the scene’s structure while later frames refine details and expand visibility.

3.3. Regularization for Stable Training

To prevent degenerate solutions, we introduce additional regularization terms:

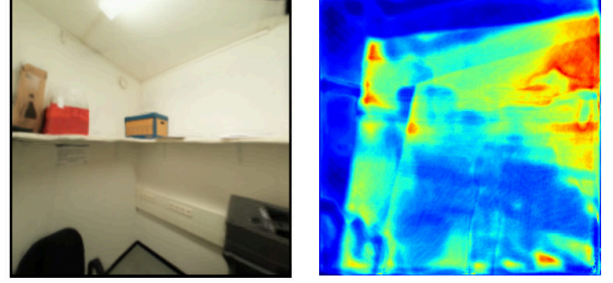


Figure 4. **Frequency Normalization.** We track how often each Gaussian contributes to the rendered pixels and normalize its gradient accordingly to ensure balanced updates.

Geometric Consistency. We impose a confidence-weighted regression loss on the predicted Gaussian means:

$$\mathcal{L}_{\text{conf}} = \sum_t \sum_{i=1}^{HW} (C_t^i \ell_{\text{regr}}(\hat{X}_t^i, X_t^i) + \alpha \log C_t^i). \quad (3)$$

This ensures that Gaussians remain anchored to surfaces rather than floating in empty space.

Opacity and Scale Regularization. To prevent redundant Gaussians for highly overlapping context frames and encourage uniform splat size, we add:

$$\mathcal{L}_{\text{opacity}} = \sum_i |\sigma_i|, \quad \mathcal{L}_{\text{scale}} = \sum_i \|\mathbf{s}_i\|_2^2. \quad (4)$$

where σ_i and $\mathbf{s}_i$ denote opacity and scale. This prevents large, unstable splats and ensures a sparse representation.

3.4. Frequency-based Gradient Normalization

Since each Gaussian may contribute to multiple pixels across different frames, we prevent over-penalization by introducing a **frequency normalization** scheme (Fig. 4). We maintain a per-Gaussian frequency map and scale each gradient update by $1/f_i$, where f_i is the usage count. This ensures:

- Splats that are frequently sampled do not dominate optimization.
- Underrepresented regions are not neglected due to fewer updates.

3.5. Final Loss Function

Our complete objective function is:

$$\mathcal{L} = \lambda_{\text{photo}} \mathcal{L}_{\text{photo}} + \lambda_{\text{temp}} \mathcal{L}_{\text{temp}} + \lambda_{\text{conf}} \mathcal{L}_{\text{conf}} + \lambda_{\text{reg}} (\mathcal{L}_{\text{opacity}} + \mathcal{L}_{\text{scale}}). \quad (5)$$

By combining photometric accuracy, geometric regularization, and frequency-aware updates, I-GaSp3R achieves stable, high-quality 3D reconstruction in real-time.

FF	PF	Method	2 views				10 views				50 views			
			PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	time	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	time	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	time
✓		MVSplat	23.67	0.811	0.180	1.63	19.31	0.697	0.334	7.42	-	-	-	-
✓	✓	Splatt3R	23.12	0.809	0.172	-	-	-	-	-	-	-	-	-
✓	✓	Spann3R	24.52	0.835	0.153	-	22.75	0.805	0.165	-	18.90	0.725	0.312	-
✓	✓	PREF3R	22.83	0.800	0.124	-	22.60	0.793	0.128	-	20.38	0.702	0.206	-
✓	✓	Ours	26.46	0.912	0.126	0.048	24.90	0.880	0.151	0.39	20.81	0.749	0.286	2.1

Table 1. **Novel-view synthesis performance on Scannet++**. All metrics are averaged over 10 validation scenes. For input views of 2, 10, and 50, the frame sampling intervals are 5, 3, and 2, respectively.

FF	PF	Method	2 views				10 views				50 views			
			PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	time	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	time	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	time
✓		MVSplat	23.00	0.782	0.210	1.63	19.43	0.694	0.357	7.42	-	-	-	-
✓	✓	Splatt3R	17.46	0.517	0.349	-	-	-	-	-	-	-	-	-
✓	✓	Spann3R	19.50	0.648	0.172	-	19.96	0.631	0.218	-	17.56	0.505	0.335	-
✓	✓	PreF3R	20.96	0.712	0.187	-	21.91	0.693	0.210	-	18.70	0.563	0.300	-
✓	✓	Ours	29.13	0.912	0.126	0.048	24.90	0.880	0.151	0.39	-	-	-	-

Table 2. **Novel-view synthesis performance on ArkitScenes**. Experimental settings and evaluation criteria are consistent with those described in Tab. 1.

4. Results

4.1. Experimental setup

Dataset We train our model on the large-scale ScanNet++ [23] dataset, which primarily comprises diverse indoor scenes annotated with ground-truth metric depth and camera poses. We utilize all the data from the official training split of ScanNet++ for training, including a total of 2,509 scenes. For evaluation, we select 10 scenes from the validation split of ScanNet++, covering various kinds of scenarios. For each evaluation scene, we randomly choose a starting frame from the original image sequence. The lengths of the sampled evaluation frame sequences are set to 2, 10, and 50, with frames sampled at intervals of 5, 3, and 2, respectively. All frames within these sampling intervals are used as ground-truth images for novel-view synthesis.

Training details Our model is trained using a progressive multi-stage strategy, where the input view overlap is dynamically reduced from 0.8 to 0.4 as training progresses, enhancing the model’s generalization ability. Due to memory constraints, training is conducted with a batch size of 1 for a total of 30 epochs. We initialize the model using pretrained weights from Spann3R.

We use the AdamW optimizer with two distinct learning rate schedules: standard parameters are trained with an initial learning rate of 1×10^{-4} , while the memory encoder utilizes a lower learning rate of 1×10^{-5} . A step decay schedule is applied, reducing the learning rate by 10% at every 10% interval of total training epochs. We incorpo-

rate a weight decay of 0.05 and apply gradient clipping at a threshold of 0.5 to stabilize training dynamics.

For the loss function, we use the following parameters: progressive reconstruction loss (1.0), causal consistency loss (0.15, progressively decaying), MSE loss (1.0), LPIPS perceptual loss (0.1), and regularization terms for opacity and scale (0.025 each). To optimize computational efficiency, we employ mixed-precision training with gradient accumulation over 10 steps. All experiments are conducted on a single NVIDIA RTX 3080 GPU.

Evaluation metrics We evaluate rendering quality using commonly employed metrics for novel-view synthesis: PSNR, LPIPS, and SSIM. Additionally, we report the overall inference time for MVSplat and I-GaSp3R to compare their efficiency.

4.2. Evaluation

In Table 1 and Table 2, we present the quantitative results on ScanNet++ and ARKitScenes, respectively, comparing I-GaSp3R with baseline models, with the specific evaluation scenes detailed in Section 8. The data is sourced from [5]. It is important to note that these results are derived from a similar, but not identical, validation dataset. Therefore, the model should be re-evaluated once the code from [5] is released. Our findings indicate that I-GaSp3R achieves state-of-the-art (SOTA) performance, particularly in the 2-frame setting. Furthermore, it demonstrates promising results for long-tail sequences. These quantitative results are further supported by the qualitative examples from ScanNet++, shown in Fig. 5.

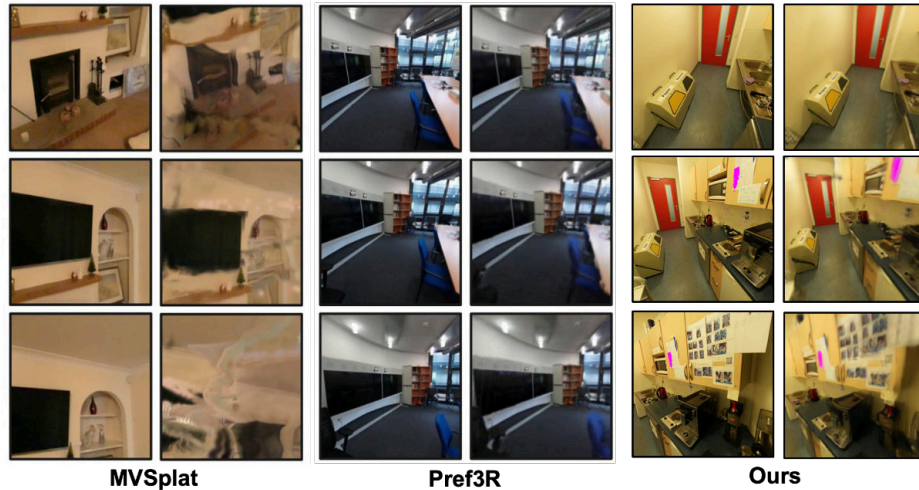


Figure 5. **Qualitative comparison of novel view synthesis.** Visualization of reconstructions from ScanNet++. Each row corresponds to a unique viewpoint, while each column displays the output of a different method

4.3. Limitations

Our model is currently trained with a context length of only 3, and only the weights of the Gaussian head and point head are fine-tuned due to GPU memory limitations. Increasing the context length and fine-tuning additional components could potentially improve performance in both short- and long-tail sequences. Furthermore, training and evaluating I-GaSp3R on datasets that include 360-degree walkarounds of objects and loop closure scenarios would better demonstrate the potential and limitations of our method.

5. Conclusion

In this work, we presented Incremental Gaussian Splatting for 3D Reconstruction (I-GaSp3R), a novel approach that integrates spatial memory with feed-forward Gaussian splatting to enable real-time, pose-free 3D scene reconstruction and novel-view synthesis. We stabilized training by the introduction of novel temporal consistency and regularization losses. Our approach marks an important step toward fast, scalable, and real-time 3D reconstruction.

References

- [1] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. In *Proc. SIGGRAPH*, 2001. 1
- [2] David Charatan, Sizhe Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction, 2024. 2, 1
- [3] Gaurav Chaurasia, Sylvain Duchene, Olga Sorkine-Hornung, and George Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM Transactions on Graphics (TOG)*, 32(3):1–12, 2013. 2
- [4] 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. *ECCV*, 2024. 2, 1
- [5] Zequn Chen, Jiezhi Yang, and Heng Yang. Pref3r: Pose-free feed-forward 3d gaussian splatting from variable-length image sequence, 2024. 4
- [6] Eric Drexheimer and Andrew J Davison. Learning a depth covariance function. In *CVPR*, pages 13122–13131, 2023. 2
- [7] Carl Doersch, Yi Yang, Mel Vecerik, Dilara Gokay, Ankush Gupta, Yusuf Aytar, Joao Carreira, and Andrew Zisserman. Tapir: Tracking any point with per-frame initialization and temporal refinement. In *ICCV*, pages 10061–10072, 2023. 2
- [8] Yilun Du, Cameron Smith, Ayush Tewari, and Vincent Sitzmann. Learning to render novel views from wide-baseline stereo pairs. In *CVPR*, 2023. 1
- [9] Arda Duzceker, Silvano Galliani, Christoph Vogel, Pablo Speciale, Mihai Dusmanu, and Marc Pollefeys. Deepvideomvs: Multi-view stereo on video with recurrent spatio-temporal fusion. In *CVPR*, pages 15324–15333, 2021. 2
- [10] Martin Eisemann, Bert De Decker, Marcus Magnor, Philippe Bekaert, Edilson De Aguiar, Naveed Ahmed, Christian Theobalt, and Anita Sellent. Floating textures. In *Computer Graphics Forum*, pages 409–418, 2008. 2
- [11] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world’s imagery. In *CVPR*, 2016. 2
- [12] Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. The lumigraph. In *Computer Graphics and Interactive Techniques*, 1996. 1
- [13] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Co-tracker: It is better to track together. *arXiv preprint arXiv:2307.07635*, 2023. 2
- [14] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r, 2024. 2

- [15] Marc Levoy and Pat Hanrahan. Light field rendering. In *SIGGRAPH*, 1996. [1](#)
- [16] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. [2](#)
- [17] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *ICCV*, 2021. [2](#)
- [18] Brandon Smart, Chuanxia Zheng, Iro Laina, and Victor Adrian Prisacariu. Splatt3r: Zero-shot gaussian splatting from uncalibrated image pairs, 2024. [1](#)
- [19] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. *TOG*, 25(3):835–846, 2006. [2](#)
- [20] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*, pages 402–419, 2020. [2](#)
- [21] Hengyi Wang and Lourdes Agapito. 3d reconstruction with spatial memory, 2024. [2](#)
- [22] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *CVPR*, pages 20697–20709, 2024. [2](#)
- [23] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *ICCV*, 2023. [4](#)

I-GaSp3R: Incremental Gaussian Splatting for 3D Reconstruction

Supplementary Material

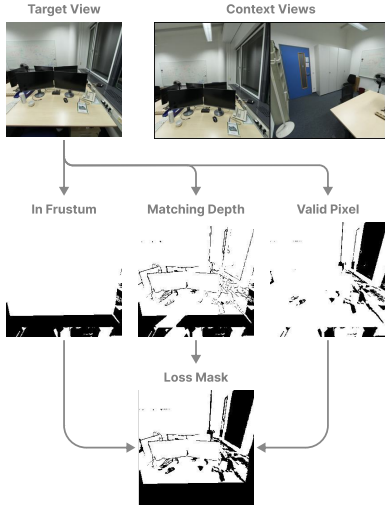


Figure 6. **Loss masking.** Valid pixels are considered to be those that are: inside the frustum of at least one of the views, have their reprojected depth match the ground truth, and are considered valid pixels with valid depth in their dataset.

6. Loss Mask

Target images may contain regions occluded or outside the context view frustums. Supervising rendering loss for such pixels is counterproductive and can degrade model performance. Existing generalizable radiance field methods mitigate this by synthesizing only interpolated view-points [2, 4, 8], minimizing unseen regions. In contrast, we train our model to extrapolate beyond input views.

To address this, we adopt the loss masking strategy of Splatt3R [18]. For each target image, we identify visible pixels by unprojecting target points and reprojecting them onto context images, verifying depth consistency. Figure 6 illustrates the mask construction.

7. Ablation Studies

Long-Tail Sequences We tested our model on long-tail sequences, observing a decreasing PSNR while still maintaining a value above approximately 21 dB after 50 context frames.

Memory Footprint The incremental nature of I-GaSp3R allows for linearly bounded memory consumption, unlike models such as MSPlat, which require quadratic memory due to processing all images at once and utilizing cross-attention. This allows our model to process image sequences of unlimited length.

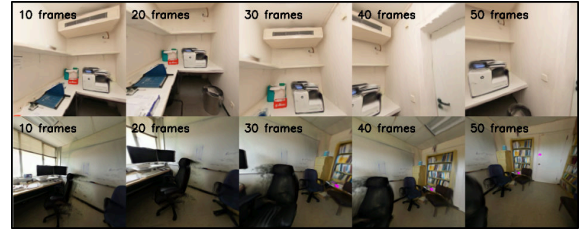


Figure 7. **Long-tail sequences.** The visualization presents rendering results of the next target view after incorporating 10 additional context views. Despite the new context views being sampled from overlapping regions, they do not disrupt the results.

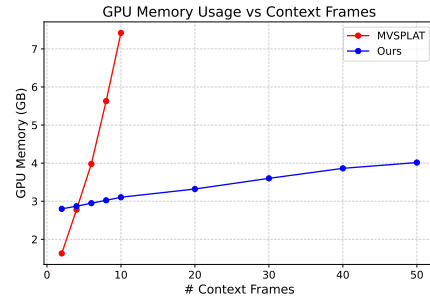


Figure 8. **Qualitative results on long-tail sequences.** After incorporating additional context views from similar viewing angles, I-GaSp3R avoids blurring existing areas by selectively adding low-opacity Gaussians, focusing on image refinement.

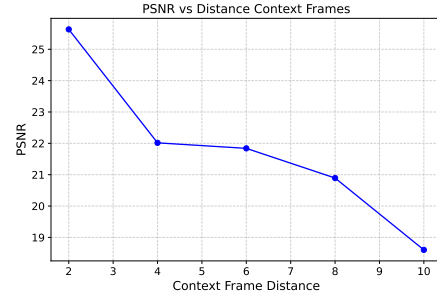


Figure 9. **Wide Baseline.** Increasing the distance between context frames, reduced the rendering results. Our model is best suited for small baseline scenarios.

Wide Baseline We tested our model with varying distances between the key frames of the input images. Increasing the distance results in a greater disparity in camera perspectives, making reconstruction more challenging. Our results show that our model performs best in small-baseline scenarios.

8. Evaluation Scenes

For the evaluation of our method, we utilized scenes from the **ScanNet++** and **ARKitScenes** datasets. The selection of these scenes was based on diverse environmental conditions and structural variations to ensure comprehensive performance assessment.

ScanNet++ Scenes The following ScanNet++ scene identifiers were included in our evaluation:

- fb5a96b1a2, 3f15a9266d, 5748ce6f01, a24f64f7fb, f9f95681fd
- f2dc06b1d2, 25f3b7a318, a980334473, e7af285f7d, c49a8c6cff

ARKitScenes The evaluation on ARKitScenes was conducted using the following scene identifiers:

- 47895783, 48018572, 48458660, 47895750, 48458465
- 48018741, 48458667, 47895745, 48018562, 48018956

These scenes were selected to cover a wide range of spatial configurations, lighting conditions, and scene complexities, providing a robust basis for evaluating the generalization capability of our approach.

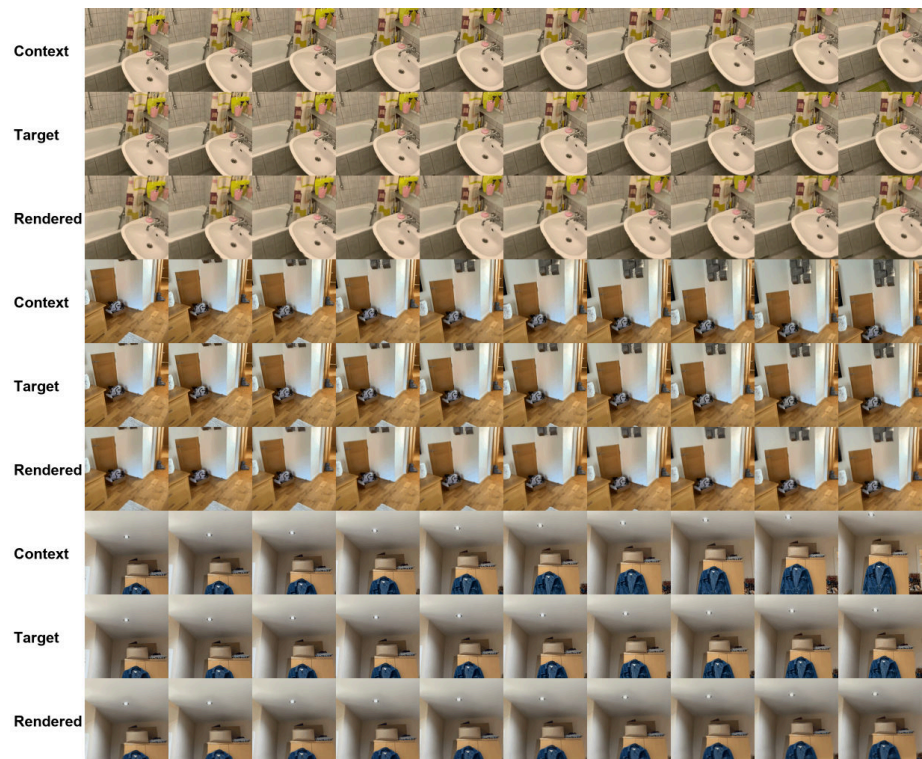


Figure 10. Results on ARKitScenes.

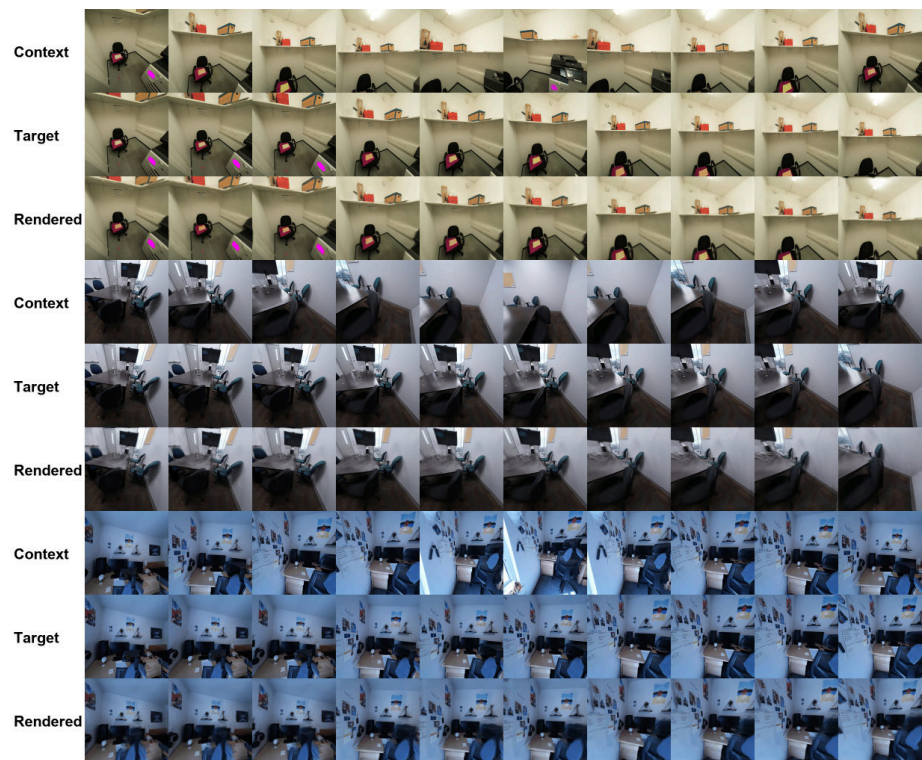


Figure 11. Results on ScanNet++.

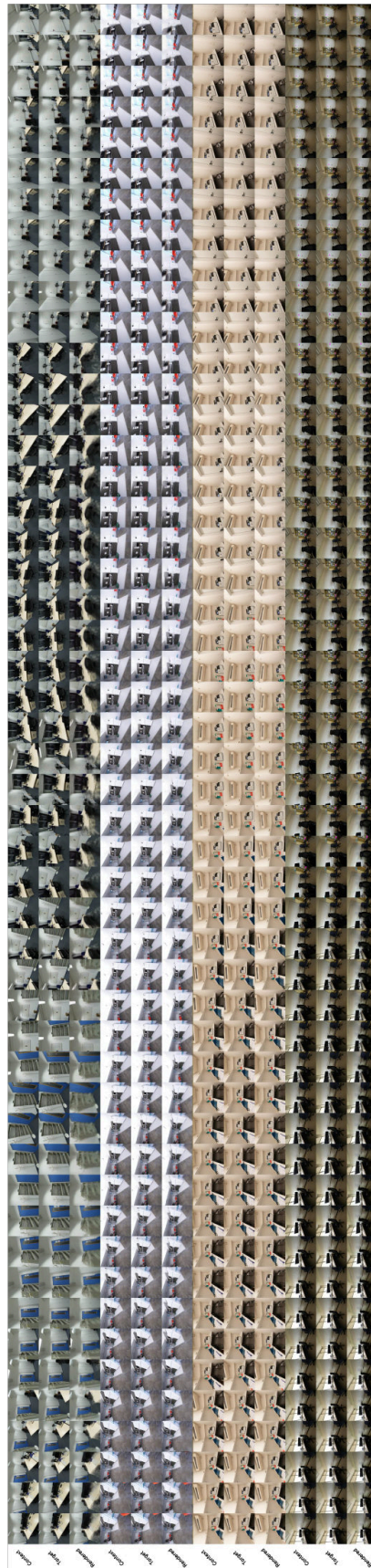


Figure 12. Long sequence results on ScanNet++.