Revisiting Depth Representations for Feed-Forward 3D Gaussian Splatting

Duochao Shi[1*], **Weijie Wang**[1, 4*], **Donny Y. Chen**[2],
Zeyu Zhang[2, 4], **Jia-Wang Bian**[3], **Bohan Zhuang**[1]

[1]Zhejiang University, China [2]Monash University, Australia [3]MBZUAI [4]GigaAI

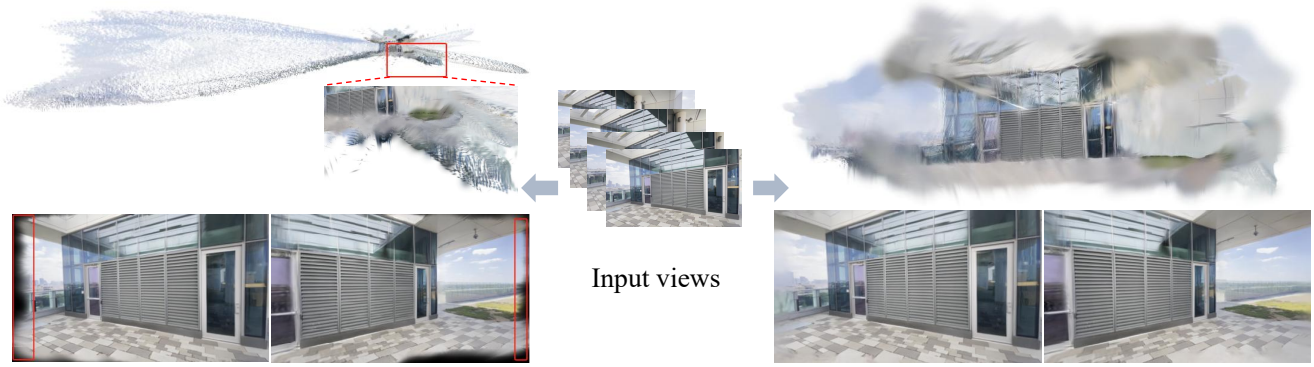

Figure 1. **PM-Loss** improves feed-forward 3DGS by using a pointmap prior to correct geometric artifacts common in depth-based approaches. Methods like DepthSplat [39] rely on unprojected depth to form 3D Gaussians. However, the inherent discontinuities of depth near boundaries often propagate into distorted 3D point clouds (top left) and degraded rendering (bottom left). Our PM-Loss effectively addresses this, achieving higher-quality geometry (top right) and rendering (bottom right).

Abstract

*Depth maps are widely used in feed-forward 3D Gaussian Splatting (3DGS) pipelines by unprojecting them into 3D point clouds for novel view synthesis. This approach offers advantages such as efficient training, the use of known camera poses, and accurate geometry estimation. However, depth discontinuities, which are particularly problematic at the boundaries of the reconstructed geometry, often lead to fragmented or sparse point clouds, degrading rendering quality—a well-known limitation of depth-based representations. To tackle this issue, we introduce **PM-Loss**, a novel regularization loss based on a pointmap predicted by a pre-trained transformer. Although the pointmap itself may be less accurate than the depth map, it provides a powerful prior for geometric coherence and structural completeness, especially at the very edges where depth prediction falters. With the improved depth map, our method significantly improves the feed-forward 3DGS across various architectures and scenes, delivering consistently better rendering results. Project page:* https://aim-uofa.github.io/PMLoss

1. Introduction

Novel view synthesis (NVS) is a long-standing topic in computer vision and graphics, recently drawing increasing attention due to advances in neural rendering, particularly 3D Gaussian Splatting (3DGS) [16]. While NVS models take 2D images as inputs and outputs, their primary goal is to recover the underlying 3D scene structure. Hence, coherent and accurate geometry is essential for generating high-quality novel views. This has led to a series of research efforts aimed at enhancing visual quality by learning more precise and consistent geometric representations [10, 13, 19, 38, 46].

Although 3DGS models have ultra-fast rendering speed, reconstructing them for unseen scenes requires a time-consuming per-scene optimization process, limiting their usability in real-world applications. This challenge has led to the development of feed-forward 3DGS methods [3, 5], the main focus of our work. Unlike per-scene tuning methods that improve visual quality by learning better geometry, feed-forward 3DGS models typically fall short in geometric quality, despite significant progress aimed at enhancing appearance [27, 36, 39, 47, 49, 53]. The core issue lies in the

representation used by feed-forward methods, which rely on *depth maps*. Most feed-forward models predict depth maps and then unproject them to form 3D Gaussians. Since depth maps often contain discontinuities, which are especially pronounced near boundaries [2, 22, 26], directly unprojecting them transfers these artifacts to the 3D representation. This results in degraded geometry quality because neural networks often fail to predict sharp depth steps, instead producing erroneous values at these boundaries. When unprojected, these inaccurate predictions manifest as fragmented floaters located incorrectly in space or as sparse gaps along reconstructed edges.

Recently, 3D reconstruction has been dominated by a new line of research that adopts a representation known as the pointmap [34]. Unlike depth maps, which represent a scalar value $d \in \mathbb{R}^1$ in camera space, pointmaps encode a set of 3D points $p \in \mathbb{R}^3$ in world space, allowing for more structurally coherent and complete modeling of geometry. In addition, pointmaps simplify the traditional multi-view stereo (MVS) [12, 43] process by reformulating it as a direct regression task through a neural network. These advantageous properties have contributed to the success of many recent feed-forward approaches to 3D reconstruction [28, 30, 31, 33, 40, 48].

The success of pointmaps in accurate and regression-based 3D reconstruction motivates us to introduce them as a strong prior to reduce artifacts in depth map based feed-forward 3DGS. This is not straightforward, as pointmaps *implicitly* encode *coarse* camera poses [34], while feed-forward 3DGS performs best with *explicitly* provided *accurate* poses [3, 5, 53], making it challenging to leverage the geometry prior effectively. Existing methods that adopt pointmap priors in feed-forward 3DGS often assume a pose-free setting [25, 44]. While this avoids the pose issue by ignoring it, novel view evaluation either relies on testing with a specific dataset already used by the pointmap model (*e.g.*, ScanNet [45] in Splatt3R [25]) or requires slow test time pose alignment (*e.g.*, NoPoSplat [44]), both of which hinder real-world usability. Although one might inject camera poses into pointmap-based feed-forward models using, for example, Plücker ray embeddings, this approach is sub-optimal as it requires expensive retraining to realign the pose distribution implicitly embedded in pointmaps and does not enhance the quality of scene details.

We propose a novel method to transfer the geometry prior from pointmap regression models to feed-forward 3DGS by formulating it as a simple yet effective training loss. Unlike prior methods [25, 44] that attach an additional "Gaussian head" to the pointmap backbone, introducing the pose dilemma and requiring customization for each specific model, our approach is plug-and-play and avoids pose issues entirely. In particular, our PM-Loss guides the learning of point clouds unprojected from predicted depth by taking the global pointmap predicted by a large-scale 3D reconstruction model, *e.g.*, Fast3R [40], VGGT [31], as a pseudo-ground truth. This guidance requires that the source and target points are in the same space and that efficient measurements are available. For the former, we find that the Umeyama algorithm can efficiently align the two point clouds (see Tab. 5), leveraging the one-to-one correspondence between depth maps and pointmaps. For the latter, the Chamfer loss is used to directly regularize them in 3D space, leading to significantly better geometry quality than those applied in 2D space (see Tab. 4). By distilling the geometry prior embedded in the pointmap predicted by a pre-trained 3D reconstruction model, our method mitigates discontinuities caused by unprojected depth and significantly boosts the quality of predicted 3D point clouds and rendered novel views for feed-forward 3DGS models—see Fig. 1.

To verify the effectiveness of our PM-Loss, we apply it to train two representative feed-forward 3DGS models, MVSplat [5] and DepthSplat [39], on two large-scale benchmarks, RealEstate10K [51] and DL3DV [18]. Experiments demonstrate that our PM-Loss improves both the quality of the 3D Gaussians and the rendered novel views across all reported metrics. Extensive ablation studies and analysis further validate architectural design choices, as well as efficient memory and runtime usage of our PM-Loss. Given its plug-and-play, efficient, and effective nature, we believe that PM-Loss will play an important role in training feed-forward 3DGS in the future. Our contributions are threefold:

- We pinpoint an unexposed yet critical issue that leads to lower-quality 3D Gaussians predicted by feed-forward 3DGS models, rooted in the long-standing discontinuity issue of depth.
- We introduce a novel training loss, PM-Loss, designed to improve 3D Gaussian quality by leveraging the geometry prior from pointmaps obtained from pre-trained 3D reconstruction models.
- Extensive experiments on existing feed-forward 3DGS models across two large-scale datasets demonstrate the effectiveness of our PM-Loss in enhancing the quality of both 3D Gaussians and rendered novel views.

2. Related Work

3D Gaussian Splatting. Novel view synthesis (NVS) techniques have evolved significantly, transitioning from traditional image-based methods [4, 24] to modern neural rendering approaches [16, 20]. Early neural methods, such as NeRF [20], represent scenes implicitly. While they can achieve high-fidelity results, they are typically hampered by slow rendering speeds and the requirement for dense input views. More recently, explicit representations like 3D Gaussian Splatting (3DGS) [16] and its subsequent vari-

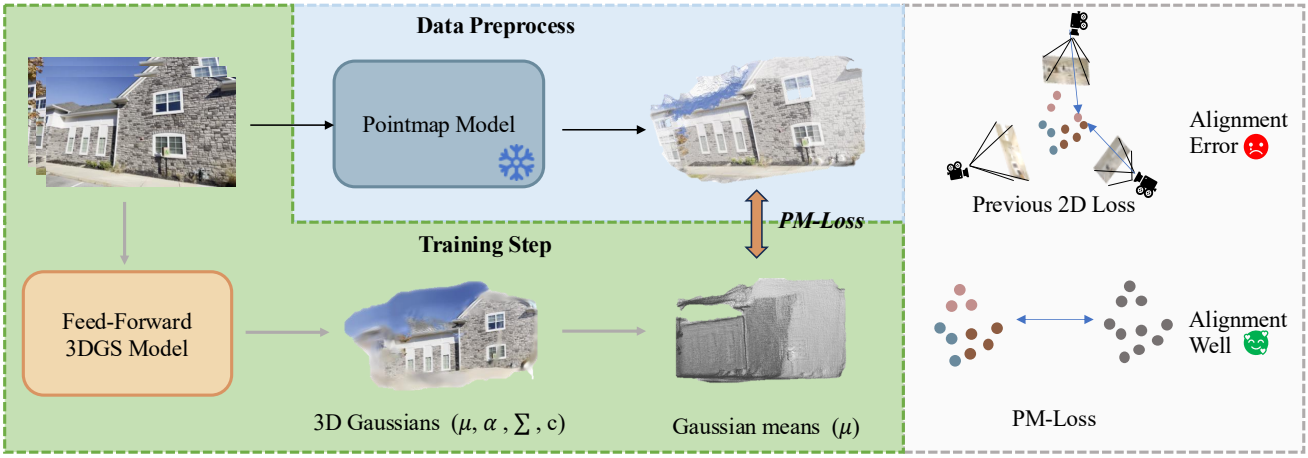

Figure 2. **Overview of PM-Loss.** The process begins by estimating a dense pointmap of the scene using a pre-trained model. This estimated pointmap then serves as direct 3D supervision for training a feed-forward 3D Gaussian Splatting model. Crucially, unlike conventional methods relying predominantly on 2D supervision, our approach leverages explicit 3D geometric cues, leading to enhanced 3D shape fidelity.

ants [10, 13, 19, 38, 46] have emerged, offering significantly faster rendering speeds due to their rasterization-friendly nature.

Several recent works have explored leveraging geometric priors to optimize the 3D Gaussian representation. Specifically, some methods [8, 52] incorporate depth information as a prior for optimizing the geometry of the Gaussians. However, a common limitation of these approaches is their reliance on monocular depth estimation models [41, 42]. Such monocular priors, especially when estimated independently per-image, often suffer from inconsistencies and multi-view misalignment, hindering geometric accuracy. Our approach mitigates this multi-view inconsistency by leveraging supervision from a multi-view consistent 3D point prior derived from pretrained models.

Feed-Forward 3D Gaussian Splatting. Moving beyond per-scene optimization, pixelSplat [3] presented a pioneering feed-forward approach for 3DGS, predicting Gaussian parameters directly from two input views with help of epipolar transformers. Subsequently, MVSplat [5] improved the efficiency of feed-forward 3DGS by proposing a cost-volume based feature fusion method, enhancing its practicality. DepthSplat [39] further explored incorporating depth priors [41, 42] into the feed-forward 3DGS framework, aiming to improve the geometric accuracy of the predictions. Similar to the previous methods [6, 8, 11, 15, 21, 35–37, 52], these feed-forward methods are often challenged by the difficulty in ensuring accurate multi-view consistency during feature processing or prior fusion, leading to geometric inaccuracies in the generated 3D Gaussian representation.

3D Reconstruction Using Pointmap. Recent advancements in 3D pointmap reconstruction methods [17, 28, 30–32, 34, 50] that produce highly accurate 3D point clouds have gained significant attention. Representative works in this area include DUSt3R [34], which utilizes a large Transformer-based model for robust multi-view feature fusion to generate dense 3D points. Building upon DUSt3R [34], MV-DUST3R [28] further extends its capability to handle an arbitrary number of input views by facilitating information exchange across them, typically considering one as a reference view. Fast3R [40] focuses on highly efficient reconstruction, demonstrating the ability to process over 1000 images in a single forward pass. VGGT [31] infers key 3D attributes of a scene by combining features from models like DINO and DPT. While these methods excel in geometric reconstruction accuracy, they are typically not designed for direct novel view synthesis and often incur significant training costs. Our method, PM-Loss, aims to bridge the gap by combining the strengths of efficient feed-forward 3DGS networks and the accurate geometric priors provided by point-map-based large models, resulting in improved geometric quality for feed-forward NVS.

3. Methodology

Our goal is to train a neural network that directly predicts a 3D Gaussian Splatting (3DGS) model from one or more input images for novel view synthesis, eliminating the need for per-scene optimization. To enhance the quality of the predicted 3D Gaussians, we introduce a novel PointMap Loss (PM-Loss) that regularizes the predicted 3D structure. PM-Loss leverages pointmaps—structured 2D-to-3D representations regressed from input images using a pretrained Vision Transformer [9]—to provide image-aligned supervi-

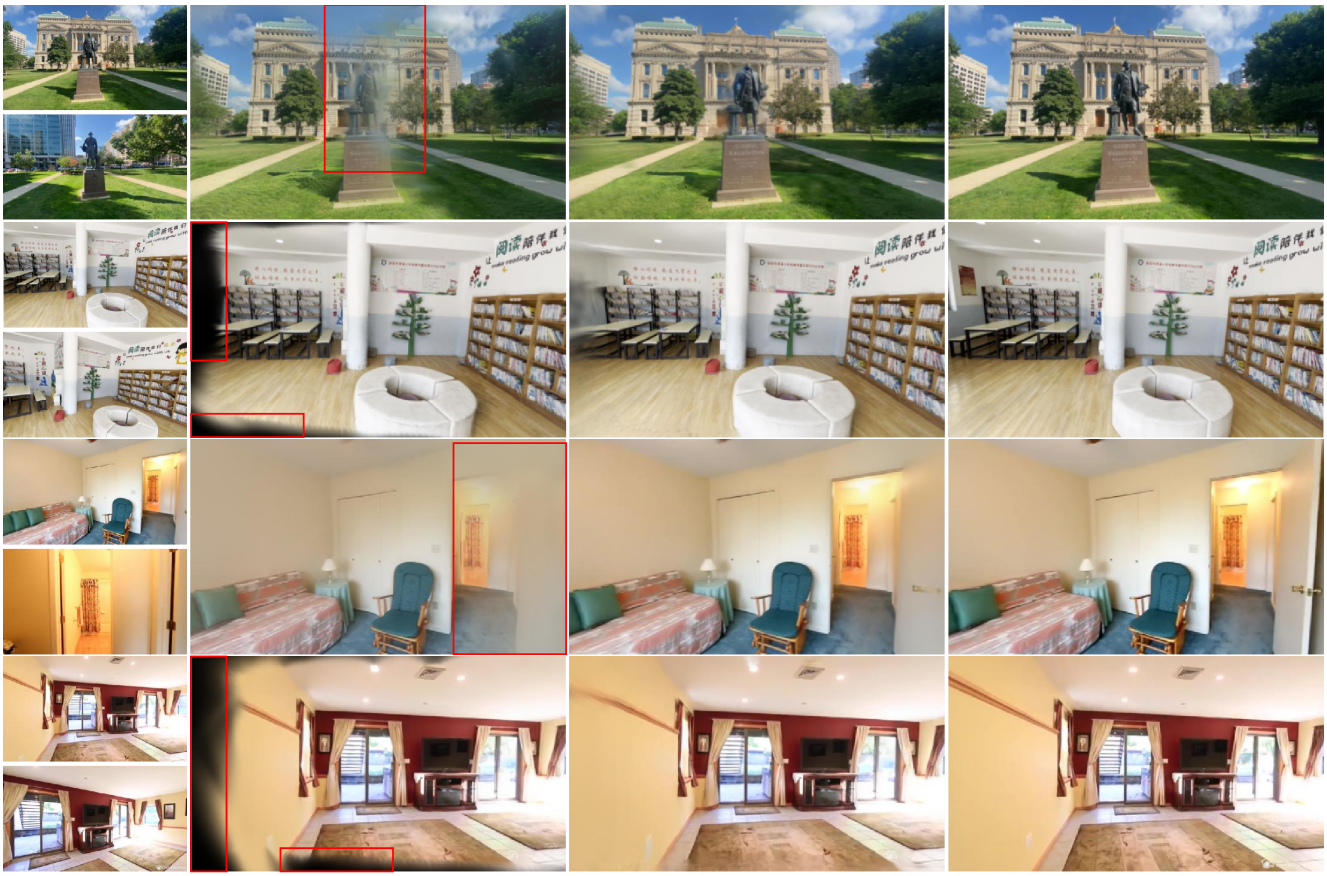

Figure 3. **Qualitative comparisons on DL3DV(top two rows) and RealEstate10K(bottom two rows).** Adding PM-Loss leads to significant improvements in rendering quality at boundaries. Note the mitigation of blurry artifacts(row 1,3) and black regions(row 2,4) in the rendered views.

sion for geometry learning. We begin by introducing the necessary preliminaries, followed by a detailed description of our PM-Loss design.

3.1. Preliminary

Feed-Forward 3D Gaussian Splatting. The method aims to reconstruct a set of 3D Gaussians directly from one or several input images in a single forward pass. The general architecture involves an encoder-decoder structure. First, an encoder network processes the input image(s) I to extract high-level features F:

$$F = \text{Encoder}(I) \tag{1}$$

These features F are then typically fused with camera pose information P_{cam} and potentially other supplementary information S_{aux} through a fusion module, $\text{Fuse}(\cdot)$. A subsequent Gaussian head network, $G_{\text{head}}(\cdot)$, then predicts the parameters for a collection of N 3D Gaussians. These parameters include the mean (center) $\mu \in \mathbb{R}^3$, covariance $\Sigma \in \mathbb{R}^{3\times3}$ (often represented by scale and rotation), opacity $\alpha \in \mathbb{R}$, and color $c \in \mathbb{R}^3$ (or spherical harmonic coefficients) for each Gaussian:

$$(\{\mu_i, \Sigma_i, \alpha_i, c_i\}_{i=1}^{N}) = G_{\text{head}}(\text{Fuse}(F, P_{\text{cam}}, S_{\text{aux}})) \tag{2}$$

In typical feed-forward 3DGS pipelines, the Gaussian means μ_i are derived by unprojecting predicted depth maps. Specifically, for each pixel (u, v) in an input image, a depth value d is predicted. This depth, along with the camera intrinsic matrix K and camera-to-world transformation matrix $T_{cw} = [R_{\text{ext}}|t_{\text{ext}}]$, is used to compute the 3D position of the corresponding Gaussian center μ_{uv}:

$$\mathbf{p}_{\text{cam}}(u, v) = d(u, v) \cdot K^{-1} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \tag{3}$$

$$\mu_{uv} = R_{\text{ext}} \cdot \mathbf{p}_{\text{cam}}(u, v) + t_{\text{ext}} \tag{4}$$

While this approach is efficient, it often suffers from geometric inaccuracies due to the discontinuities of pre-

dicted depth. These discontinuities, when unprojected, lead to fragmented or misplaced Gaussians, thereby degrading the geometric quality of the 3D scene representation and subsequently impacting the novel view synthesis quality. For instance, fragmented floaters create blurry artifacts, and sparse geometry results in black regions or incomplete structures in the rendered image—see Fig. 3.

Pointmap Regression. A pointmap is a structured 3D representation in which each pixel (u, v) of an input 2D image I is associated with a 3D point $p'_{uv} \in \mathbb{R}^3$ in world coordinates. Unlike depth maps, which provide only per-pixel Z-values, pointmaps directly represent full 3D coordinates (XYZ). They are typically regressed from images in a feed-forward manner using pretrained deep neural networks, often based on Vision Transformer (ViT) [9] architectures.

Let $\mathcal{R}_{\text{pm}}$ denote such a pointmap regression model. For each of the n_{img} input images I_j (with resolution $H \times W$) and its camera pose $P_{\text{cam},j}$, $\mathcal{R}_{\text{pm}}$ outputs a set of 3D points:

$$\{p'_{j,u,v} \in \mathbb{R}^3 \mid (u, v) \in I_j\} = \mathcal{R}_{\text{pm}}(I_j, P_{\text{cam},j}) \quad (5)$$

These per-image pointmaps are aggregated to form the global reference point cloud $X_{\text{PM}} = \{p'_k \in \mathbb{R}^3\}_{k=1}^{N_{\text{total\_pts}}}$, where $N_{\text{total\_pts}} = n_{\text{img}} \times H \times W$. This provides a dense 3D geometric prior that is leveraged in our PM-Loss.

3.2. PM-Loss

Existing methods like DepthSplat [39] use 2D-supervised depth priors to improve geometry, but this may not ensure 3D consistency. Instead, we advocate for directly regularizing geometry learning in 3D space. Given a batch of n_{img} input images of resolution $H \times W$, our feed-forward 3DGS model predicts a set of Gaussian centers, X_{3DGS}. For supervision, we use a reference point cloud, X_{PM}, derived from a pretrained pointmap regression model that provides a 3D point for each pixel. Both point clouds contain $N_{\text{total\_pts}} = n_{\text{img}} \times H \times W$ points and are formally defined as:

$$X_{\text{3DGS}} = \{\mu_k \in \mathbb{R}^3\}_{k=1}^{N_{\text{total\_pts}}}, \quad X_{\text{PM}} = \{p'_k \in \mathbb{R}^3\}_{k=1}^{N_{\text{total\_pts}}} \quad (6)$$

The two point clouds, X_{3DGS} and X_{PM}, share a natural one-to-one correspondence since each point pair (μ_k, p'_k) originates from the same source pixel. While the absolute accuracy of X_{PM} might be less than that achievable via finely-tuned depth map unprojection in well-textured regions, the pointmaps tend to exhibit better geometric coherence and structural completeness, especially at the boundaries of the reconstructed geometry. We leverage this prior as a form of 3D supervision to guide the feed-forward 3DGS model toward learning more coherent and consistent geometry.

Efficient Point Cloud Alignment. Although both the predicted Gaussian centers X_{3DGS} and the reference pointmap X_{PM} are in world coordinates, they are often misaligned in scale, rotation, and translation. This misalignment, stemming from pose errors or the pointmap model's implicit coordinate system, can create misleading gradients if uncorrected. Therefore, accurate alignment is crucial for effective supervision.

While traditional methods like Iterative Closest Point (ICP) [1] are too slow for training with dense point clouds(see Tab. 6), our key insight is that both X_{3DGS} and X_{PM} share a one-to-one correspondence with the input image pixels. This natural pixel-wise correspondence enables the use of the highly efficient Umeyama algorithm [29] to find the optimal similarity transformation (scale, rotation, and translation) in a closed form.

Given the corresponding source points $X_{\text{PM}} = \{p'_k\}_{k=1}^{N_{\text{total\_pts}}}$ and target points $X_{\text{3DGS}} = \{\mu_k\}_{k=1}^{N_{\text{total\_pts}}}$, the Umeyama algorithm finds the optimal similarity transformation (s^*, R^*, t^*) that minimizes their mean squared error:

$$(s^*, R^*, t^*) = \underset{s,R,t}{\operatorname{argmin}} \frac{1}{N_{\text{total\_pts}}} \sum_{k=1}^{N_{\text{total\_pts}}} \|sRp'_k + t - \mu_k\|^2 \quad (7)$$

We then apply this estimated transformation to the source pointmap X_{PM} to obtain the aligned pointmap $X'_{\text{PM}} = \{s^* R^* p'_k + t^*\}_{k=1}^{N_{\text{total\_pts}}}$. This alignment ensures our supervision loss is computed in a consistent coordinate frame.

Single-Directional Chamfer Loss. Given the aligned point clouds X_{3DGS} and X'_{PM}, we define the PM-Loss L_{PM} as a single-directional Chamfer distance from X_{3DGS} to X'_{PM}. This formulation ensures that, for each point in X_{3DGS}, we can efficiently identify its nearest neighbor in X'_{PM} to provide reliable geometric supervision.

Formally, the loss is defined as:

$$L_{\text{PM}}(X_{\text{3DGS}}, X'_{\text{PM}}) = \frac{1}{N_{\text{total\_pts}}} \sum_{\mu \in X_{\text{3DGS}}} \min_{p' \in X'_{\text{PM}}} \|\mu - p'\|_2^2 \quad (8)$$

This formulation effectively acts as a regularisation term, penalizing deviations of the predicted Gaussian centers from the geometry prior suggested by the pointmap. We adopt a simple mean squared error (MSE) averaged over all 3D Gaussian centers, which promotes stable training and ensures smooth gradient propagation.

A key insight of our proposed PM-Loss is to re-compute the nearest neighbor in 3D space for supervision, rather than directly relying on the natural one-to-one pixel correspondence, which degenerates to a depth loss. This design makes the supervision more robust to pose misalignments and prediction noise. We conduct ablation studies and report the quantitative results in Tab. 4.

DepthSplat DepthSplat + PM-Loss

Figure 4. **Qualitative comparison of unprojected 3D Gaussians on DL3DV dataset.** Our method effectively regularizes the 3D Gaussians, significantly reducing floating artifacts and noise that arise from inaccurate depth predictions.

DepthSplat DepthSplat + PM-Loss

Figure 5. **Qualitative comparison** of unprojected 3D Gaussians on the DTU benchmark.

Table 1. **Quantitative results in the boundary-aware setting.** Both MVSplat [5] and DepthSplat [39] show better rendering quality with the addition of PM-Loss.

Method	DL3DV [18]			RealEstate10K [51]		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
DepthSplat	18.46	0.689	0.261	20.43	0.788	0.218
DepthSplat+PM	**20.77**	**0.705**	**0.245**	**22.48**	**0.814**	**0.194**
MVSplat	16.79	0.592	0.322	19.52	0.757	0.231
MVSplat+PM	**19.25**	**0.615**	**0.291**	**22.18**	**0.787**	**0.199**

4. Experiments

4.1. Experimental Settings

Datasets. We evaluated our method using three datasets: DL3DV, RealEstate10K, and DTU. DL3DV [18] is a challenging large-scale collection comprising 10,510 real-world scenes. Following [39], we used the DL3DV-Benchmark (140 scenes) for Novel View Synthesis (NVS) testing and the remaining DL3DV scenes for training. RealEstate10K [51], comprising camera trajectories from 80,000 YouTube video clips (10M frames), is split into 67,477 training and 7,289 testing scenes. We used its test split for NVS evaluation, excluding scenes with too few views. DTU [14] features 128 scenes from controlled lab environments with ground truth models from a structured light scanner and corresponding depth maps. We assessed Gaussian splat geometric quality on 16 of these scenes, following [7] and [43].

Baselines. To evaluate our proposed method, we apply it to two representative feed-forward 3D Gaussian Splatting (3DGS) models: MVSplat [5] and DepthSplat [39]. Our experiments compare the performance of models fine-

Table 2. **Quantitative comparison on DTU with varying input numbers.** Adding PM-Loss consistently improves geometry across different numbers of input views.

Input	Method	Acc↓		Comp↓		Overall↓	
		Mean	Med.	Mean	Med.	Mean	Med.
2-view	DepthSplat	0.264	0.200	0.101	0.051	0.182	0.125
	DepthSplat+PM	**0.232**	**0.166**	**0.099**	**0.045**	**0.165**	**0.106**
4-view	DepthSplat	0.169	0.117	**0.066**	0.022	0.123	0.051
	DepthSplat+PM	**0.156**	**0.076**	0.069	**0.022**	**0.113**	**0.049**
6-view	DepthSplat	0.162	0.070	**0.048**	0.017	0.105	0.044
	DepthSplat+PM	**0.150**	**0.068**	0.053	**0.016**	**0.102**	**0.042**

tuned with PM-Loss against that of the same models fine-tuned with their original training objectives, ensuring both are evaluated in a fine-tuned state. For a fair comparison under otherwise identical conditions, both sets of models started from public pre-trained weights and were subsequently fine-tuned on the DL3DV dataset using the same batch size and total number of training iterations.

Metrics. For Novel View Synthesis (NVS) evaluation, we designed a boundary-aware setting to assess the geometric quality improvement. Specifically, target views were selected to lie adjacent to the spatial region of the context views. This approach intentionally frames the scene to make its geometric boundaries visible, thereby evaluating the quality of the reconstruction's periphery. Tab. 1 presents the results for this boundary-aware evaluation. In addition to NVS, to quantitatively assess the geometric quality of the generated 3D Gaussians, we treat the centers (μ) of all 3D Gaussians as a point cloud and compare this representation against the ground truth (GT) point clouds provided by DTU. We use three standard point cloud metrics: Accuracy (Acc), Completeness (Comp), and the Overall Chamfer Distance (Overall), all where lower values indicate better performance. For each of these metrics, we report both the Mean and Median (Med.) values, see in Tab. 2.

Implementation Details. Our method was implemented in PyTorch. The Chamfer distance computation within our PM-Loss utilizes the implementation from PyTorch3D [23]. All experiments were conducted on a single NVIDIA A100 GPU. We used the AdamW optimizer, fine-tuning each model variant (i.e., original base models and PM-Loss enhanced models) for $100,000$ iterations with a learning rate of 2×10^{-4}. For the pointmap supervision component of PM-Loss, we adopted VGGT [31] to generate pseudo ground truth, specifically using its publicly available 1B parameter model. The weighting coefficient for PM-Loss, λ_{PM}, was set to 0.005. Training and testing resolutions adhered to the original configurations of the base models: 256×448 for DepthSplat and 256×256 for MVSplat. Further training details are provided in the Appendix.

Table 3. **Ablation on pointmap supervision based on DepthSplat.** Note that w/o pointmap refers to the baseline DepthSplat model. While better pointmap models yield better results, the improvement of adding PM-Loss is consistent.

Pointmap	DL3DV [18]			RealEstate10K [51]		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
w/o pointmap	18.46	0.689	0.261	20.43	0.788	0.218
Fast3R [40]	20.51	0.690	0.257	22.43	0.812	0.197
VGGT [31]	**20.77**	**0.705**	**0.245**	**22.48**	**0.814**	**0.194**

Table 4. **Comparison of different distance measurements.** Our 3D "nearest-neighbor" Chamfer loss outperforms the 2D "one-to-one" depth loss, highlighting the benefits of 3D regularization.

Measurement	Acc↓		Comp↓		Overall↓	
	Mean	Med.	Mean	Med.	Mean	Med.
2D depth loss	0.254	0.179	**0.096**	0.048	0.175	0.114
3D chamfer loss	**0.232**	**0.166**	0.099	**0.045**	**0.165**	**0.106**

4.2. Main Results

In this section, we present detailed experimental results and analysis to verify the assumptions and effectiveness of our PM-Loss, covering rendered view quality, 3D point cloud quality, alternative design choices, memory and time efficiency of each component.

Visual Quality Improvement. By regularizing the predicted point clouds, our PM-Loss improves 3D Gaussian quality and, in turn, novel view rendering. Tab. 1 shows our PM-Loss boosts baseline performance on two large-scale datasets by over 2 dB in PSNR. We hypothesize that this improvement stems from the enhanced geometric coherence our method provides. Our PM-Loss is particularly effective in addressing the errors that arise from predicted depth discontinuities. These are most severe in challenging areas—for example, where a foreground element meets a distant background, or at the periphery of the captured views. As supported by the qualitative results in Fig. 3, the baseline's failure to handle these discontinuities leads to two types of rendering flaws: misplaced Gaussians that create blurry artifacts within the image, and sparse geometry at the scene's edge that results in black regions.

Point Cloud Quality Improvement. We qualitatively compare point cloud quality on DL3DV in Fig. 4, where our PM-Loss produces cleaner point cloud than DepthSplat, reducing noisy artifacts. For quantitative analysis on scenes with ground truth point clouds, we evaluate on the DTU benchmark. As shown in Tab. 2 and Fig. 5, our PM-Loss improves accuracy, completeness, and overall scores, confirming our qualitative findings. These improvements are

Table 5. **Timing breakdown of PM-Loss.** Our PM-Loss is efficient, with a total time of 65.0 ms for a typical operation.

Component	Time (ms)
Alignment (Umeyama)	0.9
Chamfer Distance	64.1
Total	**65.0**

Table 6. **Computation time comparison of alignment methods.** Our alignment method (Umeyama) is significantly more efficient than the commonly used ICP method.

Method	Time (ms)
Umeyama	0.9
ICP	238.3

consistent across varying numbers of input views (2 to 6), demonstrating our method's robustness.

4.3. Ablation Studies and Analysis

Impact of Varying Pointmaps. To assess how pointmap quality affects our PM-Loss, we compared our default pointmap source, VGGT, with another state-of-the-art model, Fast3R. As shown in Tab. 3, using Fast3R's pointmaps yields slightly lower performance than VGGT but still significantly outperforms the baseline. This confirms that while our PM-Loss benefits from higher-quality pointmaps, it is not tied to a specific generator architecture. We also note that VGGT's performance gain over Fast3R is larger on the more complex DL3DV dataset. We also observe a larger gain from VGGT over Fast3R on DL3DV (+0.2dB PSNR) than on RE10K (+0.05dB PSNR), indicating that our PM-Loss is particularly effective in more complex real-world scenarios.

Impact of Varying Distance Measurement. As discussed in Sec. 3.2, PM-Loss is designed to regularize geometry in 3D space to avoid strict pixel-alignment. To validate this, we compare our method against a "2D depth loss" baseline in Tab. 4. Specifically, this baseline computes the L_2 distance between the predicted point cloud X_{3DGS} and the aligned reference pointmap X'_{PM} by strictly enforcing their inherent one-to-one pixel correspondence. The results show that our 3D nearest-neighbor Chamfer loss consistently outperforms this 2D alternative on point cloud metrics. We attribute this to the fact that the nearest-neighbor search in world space provides more robust supervision against slight pose misalignments compared to rigid pixel-aligned correspondence.

Efficiency and Cost Analysis. The main computational overhead of our PM-Loss comes from point cloud alignment and Chamfer loss calculation. Pointmap generation itself (e.g., 0.3s/scene for VGGT) is an offline preprocessing step and does not add to the training time. As shown in Tab. 5, our PM-Loss's online components add only a minor overhead of 65ms for a large volume of over 450k 3D Gaussians(4 views with 256×448 resolution and pixel-aligned prediction). This efficiency is partly due to the point-to-point correspondence between the pointmap and the depth-unprojected point clouds, which allows fast alignment using Umeyama [29]. In contrast, a method like ICP [1] would take 250ms for the same number of 3D Gaussians (see Tab. 6), highlighting the benefit of our choice of using pointmap as geometry prior.

Table 7. **Memory usage of PM-Loss.** We report our max GPU memory usage during training depthsplat with VGGT-1B pointmap model.

Method	Max VRAM (GB)
DepthSplat	51.79
DepthSplat+PM	52.75

Regarding memory, we evaluate the VRAM usage when pointmap generation is handled as an offline preprocessing step. As shown in Tab. 7, our PM-Loss introduces only a minor VRAM increase of 0.96GB during training. Such a modest memory footprint is entirely acceptable and confirms the efficiency of our approach. Furthermore, our PM-Loss is a training-time-only regularization and does *not* introduce *any* additional cost during testing.

5. Conclusion

We present PM-Loss, a simple yet effective training loss that leverages geometry priors from pointmaps to improve feed-forward 3DGS. By regularizing in 3D space using global pointmaps as pseudo ground truth, PM-Loss alleviates depth-induced discontinuities near boundaries, leading to significantly improved geometry and rendering quality. Our PM-Loss can be seamlessly integrated into existing training pipelines and introduces *no* inference overhead. We believe PM-Loss offers a practical solution for training more robust and accurate feed-forward 3DGS models. Ultimately, our work validates that bridging NVS and 3D reconstruction through direct, aligned geometric supervision offers a highly efficient path to quality improvement, surpassing the limitations of purely image-based learning without requiring complex architectural changes.

Limitation. The effectiveness of our PM-Loss is bounded by the quality of the pre-trained pointmap model, as errors in the pointmap may propagate into the feed-forward 3DGS model through our loss. Leveraging stronger pointmap models from future 3D reconstruction advances is a promising direction.

References

[1] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, pages 586–606. Spie, 1992. 5, 8

[2] Stan Birchfield and Carlo Tomasi. Depth discontinuities by pixel-to-pixel stereo. *International Journal of Computer Vision*, 35(3):269–293, 1999. 2

[3] David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In *CVPR*, pages 19457–19467, 2024. 1, 2, 3

[4] Shenchang Eric Chen and Lance Williams. View interpolation for image synthesis. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, page 279–288, New York, NY, USA, 1993. Association for Computing Machinery. 2

[5] 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 *ECCV*, pages 370–386. Springer, 2024. 1, 2, 3, 6

[6] 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. *NeurIPS*, 37:107064–107086, 2024. 3

[7] Yuedong Chen, Haofei Xu, Qianyi Wu, Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai. Explicit correspondence matching for generalizable neural radiance fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025. 6

[8] Jaeyoung Chung, Jeongtaek Oh, and Kyoung Mu Lee. Depth-regularized optimization for 3d gaussian splatting in few-shot images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 811–820, 2024. 3

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 3, 5

[10] Lue Fan, Yuxue Yang, Minxing Li, Hongsheng Li, and Zhaoxiang Zhang. Trim 3d gaussian splatting for accurate geometry representation. *arXiv preprint arXiv:2406.07499*, 2024. 1, 3

[11] Xin Fei, Wenzhao Zheng, Yueqi Duan, Wei Zhan, Masayoshi Tomizuka, Kurt Keutzer, and Jiwen Lu. Pixelgaussian: Generalizable 3d gaussian reconstruction from arbitrary views, 2024. 3

[12] Xiaodong Gu, Zhiwen Fan, Siyu Zhu, Zuozhuo Dai, Feitong Tan, and Ping Tan. Cascade cost volume for high-resolution multi-view stereo and stereo matching. In *CVPR*, pages 2495–2504, 2020. 2

[13] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 conference papers*, pages 1–11, 2024. 1, 3

[14] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *CVPR*, pages 406–413, 2014. 6

[15] Gyeongjin Kang, Jisang Yoo, Jihyeon Park, Seungtae Nam, Hyeonsoo Im, Sangheon Shin, Sangpil Kim, and Eunbyung Park. Selfsplat: Pose-free and 3d prior-free generalizable 3d gaussian splatting. In *CVPR*, 2025. 3

[16] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 1, 2

[17] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *European Conference on Computer Vision*, pages 71–91. Springer, 2024. 3

[18] Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, et al. Dl3dv-10k: A large-scale scene dataset for deep learning-based 3d vision. In *CVPR*, pages 22160–22169, 2024. 2, 6, 7

[19] Xiaoyang Lyu, Yang-Tian Sun, Yi-Hua Huang, Xiuzhe Wu, Ziyi Yang, Yilun Chen, Jiangmiao Pang, and Xiaojuan Qi. 3dgsr: Implicit surface reconstruction with 3d gaussian splatting. *ACM Transactions on Graphics (TOG)*, 43(6):1–12, 2024. 1, 3

[20] 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. *Communications of the ACM*, 65(1):99–106, 2021. 2

[21] Zhiyuan Min, Yawei Luo, Jianwen Sun, and Yi Yang. Epipolar-free 3d gaussian splatting for generalizable novel view synthesis. *Advances in Neural Information Processing Systems*, 37:39573–39596, 2024. 3

[22] Michael Ramamonjisoa, Yuming Du, and Vincent Lepetit. Predicting sharp and accurate occlusion boundaries in monocular depth estimation using displacement fields. In *CVPR*, pages 14648–14657, 2020. 2

[23] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d, 2020. 7

[24] Steven M Seitz and Charles R Dyer. View morphing. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 21–30, 1996. 2

[25] Brandon Smart, Chuanxia Zheng, Iro Laina, and Victor Adrian Prisacariu. Splatt3r: Zero-shot gaussian splatting from uncalibrated image pairs. *arXiv preprint arXiv:2408.13912*, 2024. 2

[26] Libo Sun, Jia-Wang Bian, Huangying Zhan, Wei Yin, Ian Reid, and Chunhua Shen. Sc-depthv3: Robust self-supervised monocular depth estimation for dynamic scenes. *IEEE transactions on pattern analysis and machine intelligence*, 46(1):497–508, 2023. 2

[27] Stanislaw Szymanowicz, Eldar Insafutdinov, Chuanxia Zheng, Dylan Campbell, Joao F Henriques, Christian Rupprecht, and Andrea Vedaldi. Flash3d: Feed-forward generalisable 3d scene reconstruction from a single image. In *2025 International Conference on 3D Vision (3DV)*, pages 670–681. IEEE, 2025. 1

[28] Zhenggang Tang, Yuchen Fan, Dilin Wang, Hongyu Xu, Rakesh Ranjan, Alexander Schwing, and Zhicheng Yan. Mv-dust3r+: Single-stage scene reconstruction from sparse views in 2 seconds. In *CVPR*, 2025. 2, 3

[29] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):376–380, 1991. 5, 8

[30] Hengyi Wang and Lourdes Agapito. 3d reconstruction with spatial memory. In *2025 International Conference on 3D Vision (3DV)*, pages 78–89. IEEE, 2025. 2, 3

[31] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *CVPR*, 2025. 2, 3, 7

[32] Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A. Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state. In *CVPR*, 2025. 3

[33] Ruicheng Wang, Sicheng Xu, Cassie Dai, Jianfeng Xiang, Yu Deng, Xin Tong, and Jiaolong Yang. Moge: Unlocking accurate monocular geometry estimation for open-domain images with optimal training supervision. In *CVPR*, 2025. 2

[34] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *CVPR*, pages 20697–20709, 2024. 2, 3

[35] Weijie Wang, Donny Y. Chen, Zeyu Zhang, Duochao Shi, Akide Liu, and Bohan Zhuang. Zpressor: Bottleneck-aware compression for scalable feed-forward 3dgs. *arXiv preprint arXiv:2505.23734*, 2025. 3

[36] Yunsong Wang, Tianxin Huang, Hanlin Chen, and Gim Hee Lee. Freesplat: Generalizable 3d gaussian splatting towards free view synthesis of indoor scenes. *NeurIPS*, 37:107326–107349, 2024. 1

[37] Yunsong Wang, Tianxin Huang, Hanlin Chen, and Gim Hee Lee. Freesplat++: Generalizable 3d gaussian splatting for efficient indoor scene reconstruction, 2025. 3

[38] Yaniv Wolf, Amit Bracha, and Ron Kimmel. Gs2mesh: Surface reconstruction from gaussian splatting via novel stereo views. In *European Conference on Computer Vision*, pages 207–224. Springer, 2024. 1, 3

[39] Haofei Xu, Songyou Peng, Fangjinhua Wang, Hermann Blum, Daniel Barath, Andreas Geiger, and Marc Pollefeys. Depthsplat: Connecting gaussian splatting and depth. In *CVPR*, 2025. 1, 2, 3, 5, 6

[40] Jianing Yang, Alexander Sax, Kevin J Liang, Mikael Henaff, Hao Tang, Ang Cao, Joyce Chai, Franziska Meier, and Matt Feiszli. Fast3r: Towards 3d reconstruction of 1000+ images in one forward pass. In *CVPR*, 2025. 2, 3, 7

[41] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *CVPR*, 2024. 3

[42] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. In *NeurIPS*, 2024. 3

[43] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *ECCV*, pages 767–783, 2018. 2, 6

[44] 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 *ICLR*, 2025. 2

[45] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *ICCV*, pages 12–22, 2023. 2

[46] Mulin Yu, Tao Lu, Linning Xu, Lihan Jiang, Yuanbo Xiangli, and Bo Dai. Gsdf: 3dgs meets sdf for improved neural rendering and reconstruction. *Advances in Neural Information Processing Systems*, 37:129507–129530, 2024. 1, 3

[47] Chuanrui Zhang, Yingshuang Zou, Zhuoling Li, Minmin Yi, and Haoqian Wang. Transplat: Generalizable 3d gaussian splatting from sparse multi-view images with transformers. In *AAAI*, pages 9869–9877, 2025. 1

[48] Junyi Zhang, Charles Herrmann, Junhwa Hur, Varun Jampani, Trevor Darrell, Forrester Cole, Deqing Sun, and Ming-Hsuan Yang. Monst3r: A simple approach for estimating geometry in the presence of motion. In *The Thirteenth International Conference on Learning Representations*. 2

[49] 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 *ECCV*, pages 1–19. Springer, 2024. 1

[50] 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 *CVPR*, 2025. 3

[51] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: learning view synthesis using multiplane images. *ACM Transactions on Graphics (TOG)*, 37(4):1–12, 2018. 2, 6, 7

[52] Zehao Zhu, Zhiwen Fan, Yifan Jiang, and Zhangyang Wang. Fsgs: Real-time few-shot view synthesis using gaussian splatting. In *ECCV*, 2024. 3

[53] Chen Ziwen, Hao Tan, Kai Zhang, Sai Bi, Fujun Luan, Yicong Hong, Li Fuxin, 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*, pages 4349–4359, 2025. 1, 2