Geometrically Consistent Generalizable Splatting

Anonymous Author(s)
Affiliation
Address
email

Abstract

Gaussian splatting has emerged as the preferred 3D scene representation due to its incredible speed and accuracy in novel view generation. Various attempts have thus been made to adapt multi-view structure prediction networks to directly predict per-pixel 3D Gaussians from images. However, most work has focused on enhancing self-supervised depth prediction networks to estimate additional parameters for 3D Gaussians – orientation, scale, opacity, and appearance. We show that optimizing a view-synthesis loss alone is insufficient to recover geometrically meaningful splats in this simple manner. We systematically analyse and address the inherent ambiguities in learning 3D Gaussian splats with self-supervision to learn pose-free generalisable splatting. Our approach achieves state-of-the-art performance in (i) geometrically consistent reconstructions, (ii) relative pose estimation between images, and (iii) novel-view synthesis on the RealEstate10K and ACID datasets. We also showcase zero-shot capabilities of the proposed generalizable splatting on ScanNet, where our method substantially outperforms the prior art in recovering geometry and estimating relative pose.

1 Introduction

3D Gaussian splatting (3DGS) [22] has recently revolutionized 3D structure and appearance modeling from multi-view images. Departing from traditional depth or point cloud representations of the scene structure, 3D Gaussians implicitly model surface reflections and environment lighting to encode view-dependent scene appearance. They are memory-efficient compared to explicit volumetric alternatives, and they facilitate rendering of the scene from arbitrary viewpoints in a fraction of a second. Due to these capabilities, 3D Gaussians have become a prevalent choice for scene representation.

Learning-based structure estimation methods, such as single- or two-view depth predictors, are increasingly being adapted to directly predict 3D Gaussians using feedforward neural networks. Various laudable attempts have been made recently in training neural networks to predict 3D Gaussians directly from images, achieving photorealistic results without per-scene optimization [2, 3, 43, 47, 35, 54, 51]. These methods are commonly referred to as *generalizable Gaussian splatting*. Most generalizable Gaussian splatting methods adapt well-studied one- or multi-view structure prediction networks [46, 42, 25] to estimate locations of the 3D Gaussians. These networks typically use image encoders that take in one or multiple images followed by decoders that predict Gaussian means, in the form of per-pixel depth [46] or 3D point locations [42, 25] for each input view. Nearly all generalizable Gaussian splatting methods append additional decoders to depth or point-cloud estimation architectures to predict Gaussian properties such as orientation, scale, opacity and view-dependent color – typically without much foresight. These networks are usually trained by minimizing view-synthesis loss on a few target views, closely following existing self-supervised depth estimators – though they differ in image formation due to the underlying change in scene representation.

This prevalent setup overlooks several key issues inherited from the underlying 3DGS optimization:

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

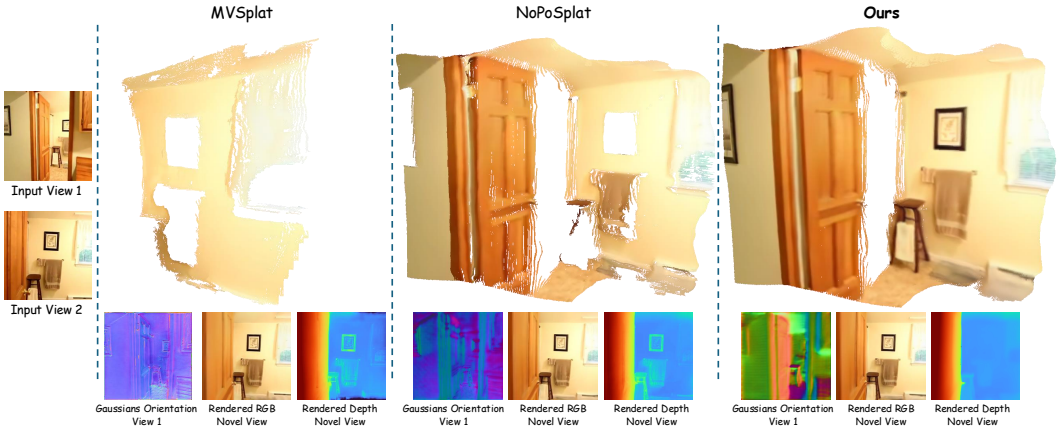

Figure 1: **Qualitative comparison of mesh reconstructions from two input views.** We compare the baseline methods MVSplat [3] and NoPoSplat [51] with our approach on RE10K dataset [57]. The *top row* displays the textured meshes reconstructed by fusing *virtual* depth maps via TSDF; the *bottom row* visualizes Gaussian-surface normals for the first input view, and the RGB/depth rendered from a novel (virtual) viewpoint. Inaccuracy in rendered depth and normals is evident for both baselines. These inconsistent depths, when fused, create several holes in the mesh reconstruction. Our method recovers accurate geometry of fine details such as the towel, wall painting, and stool.

- 3D Gaussians are grossly overparameterized compared to depth maps or point clouds. Successful estimation of 3D Gaussians typically requires a large number of densely sampled viewpoints. Few-shot 3DGS is an active field of research and often relies on regularization priors [59, 4], which are largely ignored when training generalizable splatting networks.
- Unlike per-pixel depths or 3D point locations – which are uniquely defined (up to scale) – multiple 3D Gaussian configurations can produce equally valid renderings. This inherent ambiguity makes training difficult, even when depth data is available for supervision.
- Successful per-scene Gaussian splatting methods typically rely on multiple non-differentiable heuristics (*i.e.*, splitting, and duplication of Gaussians). However, existing generalizable methods are trained purely via view-synthesis gradient loss and neglect these heuristics. i.e. they assume that all Gaussians remain perpetually alive during training.

As a result, existing generalizable Gaussian splatting methods often converge to *geometrically degenerate Gaussians*. While the predicted locations (means) remain relatively stable —- benefiting from well-established single- or multi-view depth estimators – other parameters (opacity, orientation, scale) are prone to collapse. As shown in Figure 2, existing generalizable approaches struggle to learn meaningful opacities, orientations, or scales when trained with view-synthesis loss alone in both pose-aware and pose-free settings. In particular, we observe implausible Gaussian orientations (in the form of normals) as well as unjustified elongation of the 3D Gaussians (scales).

We show that these artifacts are due to the inherent over-parametrization of geometry in the form of splats, which require structural consistency priors to make the self-supervised learning viable. By introducing such priors, our proposed method produces Gaussians that exhibit consistent and physically plausible geometric patterns. As shown in Figure 2, our proposed method produces accurate surface normals directly from the predicted Gaussian orientations. The resulting Gaussians – parameterized as 2D disks in 3D space – are elongated along geometric discontinuities and remain robust to image textures. The two approaches we selected for visualizing predicted 3D Gaussians broadly represent distinct underlying representations for encoding Gaussian means: (i) per-pixel depth maps [3], and (ii) per-pixel 3D locations aligned to a common reference frame [51]. Despite this difference, to our knowledge, all existing self-supervised generalizable splatting methods suffer from similar limitations – stemming from their reliance on the representations and loss functions introduced in [3, 51].

In this work, we aim to systematically define the ideal configuration of a geometrically consistent Gaussian and propose appropriate priors to assist generalized Gaussian splatting. To that end, we opt to build upon the recently proposed NoPoSplat framework [51] as our baseline. We choose

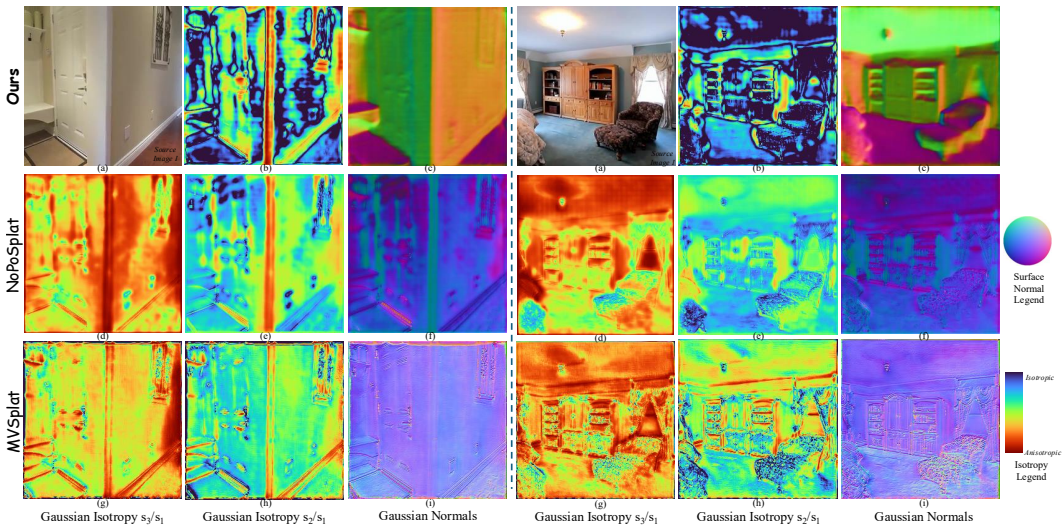

Figure 2: **Qualitative comparison of predicted Gaussian parameters**. Each Gaussian has scales $s_1 > s_2 > s_3$, where s_3 quantifies the uncertainty in localization of the surface the Gaussian belongs to with surface orientation defined by its normal [18]. **Row 1 (ours)** shows: (a) the source image to which Gaussians are aligned, (b) skewness of the estimated Gaussians within their own defining plane, and (c) predicted Gaussian orientations visualized as surface-normal maps. **Rows 2 and 3** show results for NoPoSplat and MVSplat, respectively: (d/g) Gaussians' elongation perpendicular to the dominant plane defined by it, (e/h) Gaussians' skewness within the dominant plane, and (f/i) normals to the dominant plane. Existing methods yield Gaussian orientations without clear geometric meaning: MVSplat Gaussians (i) align mostly fronto-parallel to the source image plane, and NoPoSplat Gaussians orientations (f) strongly depend on texture, spanning a few dominant directions inconsistent with scene geometry. Our method produces plausible, near-Manhattan structured surface orientations. Baseline Gaussians exhibit significant elongation perpendicular to their dominant surfaces (visible as non-red colors in d/g). Notably, our Gaussians remain relatively circular (blue color in b) on planar, textureless surfaces and become skewed ellipses (red color in b) near sharp geometric edges such as shelves or wall corners.

NoPoSplat not only for its state-of-the-art performance across relative camera pose estimation and view-synthesis but also for its self-supervised formulation, which does not require groundtruth depth maps. Additionally, utilizing DUSt3R [42] framework, the approach is one of the few to provide generalizable splatting from a pair of images without requiring the relative pose of these images.

We address ambiguities in learning over-parametrized 3D Gaussians by adding suitable regularization terms to the traditionally used view-synthesis loss. Our main observations are:

- Defining the ideal 3D Gaussian orientations to be dominant normals of the scene surfaces (as in [18]) helps resolve structural ambiguities to enable learning of the Gaussian orientations.
- Ensuring that the 3D Gaussians are pixel-aligned is extremely important with self-supervision. Particularly pose-free methods that use PnP for relative pose estimation require pixel alignment to ensure accurate camera pose and geometry prediction.
- Standard priors used to enforce consistency in rendered depth and normal maps [18] are not easily deployable for joint learning of pose and structure. Instead, we enforce consistency between Gaussian orientations and means by leveraging the local image neighborhood of pixel-aligned Gaussians. This promotes stable training in generalizable splatting networks.

We build upon NoPoSplat [51], integrating geometric consistency priors, and trained our network on the RealEstate10K (RE10K) dataset [57]. Our method outperforms prior work in novel-view synthesis and, importantly, produces plausible scene geometry that enables direct depth rendering from arbitrary viewpoints – something current methods cannot achieve. These consistent virtual depths can be fused using Truncated Signed Distance Function (TSDF) [55] and the reconstructed meshes are visualized for comparison with prior art in Figure 1. Our approach also establishes a new

state-of-the-art in relative pose estimation from image pairs, surpassing methods with task-specific training such as RoMa [6], geometry-supervised approaches such as DUSt3R [42, 25], and pose-free generalizable splatting method [51] – despite using less data and weaker supervision in some cases. Our approach achieves state-of-the-art zero-shot 3D reconstruction on ScanNet [5], outperforming all existing pose-free and pose-required generalizable splatting methods.

To the best of our knowledge, this is the first work in the domain of generalizable Gaussian splatting which systematically analyze and evaluate the veracity and geometric meaningfulness of predicted Gaussian orientations and elongations. We address the shortcomings of existing approaches in training generalizable splatting networks to produce Gaussians that enable accurate depth rendering from virtual views. We believe that the presented analysis lays the groundwork for future research on training neural networks to predict Gaussions form images, in both depth-supervised and self-supervised setups.

2 Related Work

Owing to the state-of-the-art real-time view synthesis performance of 3D Gaussian splitting [22], significant effort has been put into improving 3DGS for scenarios such as few-view reconstruction [59, 4, 26, 16, 45, 39], dynamically moving objects [44, 49, 48, 29], surface extraction [15, 18], and incorporating object semantics into 3D reconstructions [27]. Real-time simultaneous localization and mapping approaches have also adapted Gaussian splats as an inherent scene representation [31, 21]. Additionally, Gaussian splats have been used for generating geometrically consistent images and video sequences [41, 44].

The deep learning revolution of the last decade has significantly influenced geometric inference from one or more images. Earlier works focused on training neural networks to map a single image to depth map obtained from range sensors [8, 23, 9, 24, 33]. Multi-view extensions for these supervised learning algorithms are well explored as well [1, 19, 50, 14, 40, 38]. More recently, methods have explored reconstructing registered sets of per-pixel point clouds from multiple images, providing state-of-the-art relative pose and scene structure [42, 25].

Additionally, it has been demonstrated that these feed-forward geometry prediction networks can be trained without depth sensors in a self-supervised manner by minimizing view synthesis losses [11, 12, 56, 53, 13]. Structure prediction from single or few images has also been utilized as an optimization-free building block in high-fidelity tracking and mapping systems [58, 60]. Generalizable Gaussian Splatting methods have evolved recently to learn neural networks that predict 3D Gaussians explaining a scene directly from a few images. We broadly categorize these methods into following two categories:

Pose-Dependent Generalizable 3DGS: Several works assume input images come with known or pre-computed poses (e.g., via SfM) and focus on designing architectures to infer 3D Gaussians from these posed views [2, 3, 32, 43, 47, 10, 37, 54]. A prominent example is pixelSplat [2], which introduced a two-view feed-forward network that utilizes epipolar cross attention transformer architecture to fuse multi-view information and predict per-pixel depth distribution for input images. This distributions are sampled to create a set of 3D Gaussian centers along the viewing rays. MVSplat [3] uses cost volume based fusion of multi-view information, adapting the Unimatch [46] architecture to regress for depth instead. Both methods use additional decoder heads to estimate rest of the 3D Gaussian parameters.

Pose-Free Generalizable 3DGS: An emerging frontier involves dispensing with known camera poses—allowing the network to infer scene geometry and camera registration jointly from images alone [35, 20, 51]. Early efforts in this direction often build upon learned stereo matching. For example, [35] tackles uncalibrated stereo pairs by extending a foundation model (MASt3R [25]) that predicts dense point clouds from two images. It then outputs 3D Gaussians directly in a canonical frame, augmenting each point in the MASt3R reconstruction with color and covariance attributes. This process is supervised using the geometry of the 3D point cloud and followed by a novel-view synthesis stage to fine-tune appearance. NoPoSplat [51] adopts a more self-supervised, multi-view approach by anchoring one view's coordinate system as canonical and training a network to predict all Gaussians directly in that space, using only a photometric loss for training.

To the best of our knowledge, all aforementioned generalizable splitting methods struggle to learn geometrically faithful orientations and scales for 3D Gaussians. The proposed approach alleviates this issue from generalizable splatting using appropriate geometric priors.

3 Method

In this section, we present our generalizable Gaussian splatting framework and loss functions we propose to address the ill-posed nature of self-supervised learning in predicting geometrically consistent Gaussians. For the architectural details, we refer the reader to supplementary material.

Problem Definition Assuming that we are given a set of sparse images $\mathcal{I} = \{\boldsymbol{I}_t \in \mathbb{R}^{H\times W\times 3}\}_{t=1}^{T}$ (which is also known as context images in [2, 35, 3, 51, 47, 20]), each with known camera intrinsics that form the set $\mathcal{K} = \{\boldsymbol{K}_t \in \mathbb{R}^{3\times 3}\}_{t=1}^{T}$ capturing a *rigid* scene, our aim is to learn a feedforward neural network f_Θ that maps these images and intrinsics $(\mathcal{I}, \mathcal{K})$ to a set of *pixel-aligned* Gaussians as

$$f_{\boldsymbol{\Theta}}(\mathcal{I}, \mathcal{K}) = \left\{\mathcal{G}_t^j := \left(\boldsymbol{\mu}_t^j, \alpha_t^j, \boldsymbol{q}_t^j, \boldsymbol{s}_t^j, \boldsymbol{c}_t^j\right)\right\}_{t=1:T}^{j=1:H\times W}, \tag{1}$$

where $\mathcal{G}_t^j$ is the 2D Gaussian defined in the 3D space corresponding to a pixel j in image t. Each $\mathcal{G}_t^j$ is characterized by its *center* $\boldsymbol{\mu} \in \mathbb{R}^3$; *orientation* represented by a unit quaternion vector $\boldsymbol{q} \in \mathbb{R}^4$; two *scale* parameters $\boldsymbol{s} \in \mathbb{R}^2$ defining the elongation of the 3D Gaussians; *opacity* $\alpha \in \mathbb{R}$; and *color* encoded as spherical harmonics $\boldsymbol{c} \in \mathbb{R}^d$. In this work, we advocate the use of 2D Gaussians [18] to represent the scene instead of the standard 3D Gaussians adopted by prevalent generalizable Gaussian splatting frameworks [2, 3, 47, 51]. Following [18], we assume that estimated Gaussians are aligned with the scene surface and its elongation perpendicular to the local surface normal is zero. We show through extensive evaluations how this choice helps generalizable Gaussian splatting in Section 4.

Note that both Gaussian centers $\boldsymbol{\mu}_t^j$ and orientations $\boldsymbol{q}_t^j$ are defined in the image coordinates of the *first* image $\boldsymbol{I}_1$. Given these $M \times N \times T$ Gaussians predictions, we render *novel views* of the scene $\{\hat{\mathbf{I}}_f \in \mathbb{R}^{H\times W\times 3}\}_{f=1}^{F} \subset \mathcal{I}$ from F different viewpoints defined by its projection matrix $P_f = (\mathbf{R}_f, \mathbf{T}_f) \in$ SE(3) to be matched with its observed images $\mathbf{I}_f$s during training.

We propose to minimize the view synthesis loss [51, 3, 2] from the predicted Gaussians as

$$\mathcal{L}_{synthesis} = \sum_{f=1}^{F} \mathcal{L}_{rgb}(\mathbf{I}_f, \hat{\mathbf{I}}_f) + \mathcal{L}_{lpips}(\mathbf{I}_f, \hat{\mathbf{I}}_f), \tag{2}$$

where $\hat{\mathbf{I}}_f(u, v)$ is the color corresponding to a pixel (u, v) in image $\mathbf{I}_f$ rendered by blending K ordered *projected* Gaussians $\mathcal{G}'$ using the 2DGS rasterizer as

$$\hat{\mathbf{I}}_f(u, v) = \sum_{k=1}^{K} \boldsymbol{c}_k \alpha_k \mathcal{G}'(u, v) \Pi_{j=1}^{k-1}(1 - \alpha_j \mathcal{G}'(u, v)), \tag{3}$$

Note that $\mathcal{G}'$ is the projection of the Gaussians $\mathcal{G}$ onto the 2D image plane of the image $\mathbf{I}_f$, see supplementary material for more details.

As shown in Section 4, solely relying on view synthesis loss is proven to be insufficient for learning geometrically meaningful Gaussians. In this work, we propose to minimize two additional regularization losses: (i) a depth-surface normal consistency term $\mathcal{L}_{orient}$ to align the orientations of the Gaussians with the rendered depth; (ii) a grid alignment loss $\mathcal{L}_{align}$ to ensure that the estimated Gaussians are aligned with the pixels of the provided images. Combining these two regularization with the view synthesis loss, we define our training objective function $\mathcal{L}_{total}$ as

$$\mathcal{L}_{total} = \mathcal{L}_{synthesis} + \lambda_o \mathcal{L}_{orient} + \lambda_a \mathcal{L}_{align}, \tag{4}$$

where λ_o and λ_a are weighting factors balancing the influence of each regularization. We discuss the motivation, formulation and impact of the regularization term in the following sections.

3.1 Learning Gaussian's Orientations.

Recall that existing pose-free and pose-aware generalized Gaussian splatting approaches struggle to learn meaningful Gaussian's orientations, see Figure 2. To provide geometric meaning for the

orientations of the Gaussians, we propose to align them with the dominant surface normals of the scene they belong to. To that end, we follow the setup in [18] to estimate only *two non-zero Gaussian scales* $(s_t^{j,1}, s_t^{j,2})$ and set the third $s_t^{j,3}$ (along the normal) to zero, so each Gaussian is "flat" in one direction. The resulting rank-deficient Gaussian covariance matrix $\mathbf{\Sigma}_t^j$ is defined as

$$\mathbf{\Sigma}_t^j = \mathbf{R}(\boldsymbol{q}_t^j)\,\mathrm{diag}([s_t^{j,1}, s_t^{j,2}, 0]^T)\,(\mathbf{R}(\boldsymbol{q}_t^j))^T, \text{where } \mathbf{R}(\boldsymbol{q}_t^j) \in \mathrm{SO}(3) \tag{5}$$

whose null space – the zero-eigenvalue direction – encodes the Gaussian's surface normal that is predicted by the network $\boldsymbol{N}_t^j$.

Swapping 3D Gaussians to 2D surface elements reduces the over-parameterization to an extent; however, the view synthesis loss of eq. (3) does not provide a sufficiently strong supervision signal for learning orientations. The most natural way to supervise Gaussian orientation is to use surface normal regularization prior from [18], where the authors propose to enforce consistency between rendered normal and rendered depth maps while doing splatting. Naively deploying such regularization to train our model does not work well. In the supplementary material, we discuss our observations and provide remedies for successfully adapting such regularization.

Instead, we propose to use a simple yet effective alternative to supervise the predicted Gaussian orientation $\boldsymbol{q}_t^j$. Leveraging the assumption that each Gaussian $\mathcal{G}_t^j$ is aligned with the 2D image pixel $j = (u, v)$ in image t, we define the *local surface normal* for $\mathcal{G}_t^j$ using the 3D positions of its neighboring pixels as $\hat{\boldsymbol{N}}_t^j$. We enforce these estimated local surface normals to be consistent with the predicted normal $\boldsymbol{N}_t^j$ (null space of the convenience matrix $\mathbf{\Sigma}_t^j$) by minimizing the following loss

$$\mathcal{L}_{orient} = \frac{1}{T(H-2)(W-2)} \sum_{t=1}^{T} \sum_{u=2}^{W-1} \sum_{v=2}^{H-1} \|1 - < \boldsymbol{N}_t^j, \hat{\boldsymbol{N}}_t^j > \|_\rho, \tag{6}$$

$$\hat{\boldsymbol{N}}_t^j = \|(\boldsymbol{\mu}_t^{(u+1,v)} - \boldsymbol{\mu}_t^{(u-1,v)}) \times (\boldsymbol{\mu}_t^{(u,v+1)} - \boldsymbol{\mu}_t^{(u,v-1)})\|_*, \tag{7}$$

where, $< ., . >$ and $., .\times .$ are dot and cross-product or two vectors, $\|.\|_*$ represents vector normalization and $\|.\|_\rho$ is Huber loss (implemented as SmoothL1Loss in Pytorch).

Note that, unlike the loss used in 2D Gaussian splatting (2DGS) [18], $\mathcal{L}_{orient}$ does not involve any rasterization, providing a direct supervision for orientation given the Gaussian means. In fact, the proposed loss mimics the standard loss used for supervised learning of surface normals [7], where the ground truth normals are estimated from the depths using eq. (7). This simple loss in our experiments outperforms alternatives and can be used for depth supervised training of generalizable splats as well.

3.2 Pixel-aligned Gaussians

Although the first generalizable splatting approach [2] worked in a pose-aware setup and adapted two-view depth prediction network, they by construction constrains every Gaussians to lie on its corresponding viewing ray. Pose-free variants [51] drop the camera pose assumption by directly estimating the Gaussian's locations in the canonical space using a DPT decoder. While this removes the need to warp Gaussians with known cameras, the parametrization rendered the structure estimation problem ill-posed especially under self-supervised regime. Specifically, in contrast to depth-supervised frameworks like DUSt3R [42], which learns an implicit structural prior by enforcing the reconstructed 3D point cloud to project onto the regular image grid, the view synthesis loss in eq. (3) does not offer such constraint. Gaussians can therefore move freely into geometrically degenerate configurations, hampering both structure and relative pose estimation.

Therefore, we explicitly align each Gaussians to with its pixel's viewing ray. Specifically, for each pixel (u, v) in frame t, the Gaussian's centers $\boldsymbol{\mu}_t^{(u,v)}$ must be projected to that pixel location with known camera extrinsics $(\mathbf{R}, \mathbf{T})$ and intrinsic matrix $\mathbf{K}$. We enforce this with the alignment loss as

$$\mathcal{L}_{align} = \frac{1}{\sum_{t,u,v} \mathcal{M}_t^{(u,v)}} \sum_{t=1}^{T} \sum_{u=1}^{W} \sum_{v=1}^{H} \mathcal{M}_t^{(u,v)} \|[u, v]^T - \Pi(\mathbf{K}_t[\mathbf{R}_t|\mathbf{T}_t]\boldsymbol{\mu}_t^{(u,v)})\|_2^2; \tag{8}$$

where $\mathcal{M}_t^{u,v}$ is 1 if the Gaussian projects inside the image and have positive depths (otherwise 0) and $\Pi([X, Y, Z]^T) = [X/Z, Y/Z]^T$ is the perspective projection function. We demonstrate in Section 4 that the proposed loss plays a crucial role in PnP-based relative pose estimation (Table 1) as well as accurate structure estimation (Table 3).

Table 1: **Pose estimation (AUC) at multiple error thresholds on RE10K [57] (in-domain) and on ScanNet-V1 [5] and ACID [30] (cross-domain).** The overall best results are shown in **bold**, and the best result-whether with or without photometric optimization-is underlined in each section. Methods marked with † are trained on additional data (e.g., ScanNet, ACID), and those marked with ‡ use extra supervision (e.g., ground-truth depth).

	Method	RE10K			ScanNet-V1			ACID		
		5° ↑	10° ↑	20° ↑	5° ↑	10° ↑	20° ↑	5° ↑	10° ↑	20° ↑
	CoPoNeRF†	0.161	0.362	0.575	-	-	-	0.078	0.216	0.398
	DUSt3R†‡	0.301	0.495	0.657	0.085	0.210	0.398	0.166	0.304	0.437
	MASt3R†‡	0.372	0.561	0.709	0.083	0.200	0.381	0.234	0.396	0.541
	RoMa†‡	0.546	0.698	0.797	**0.168**	**0.361**	**0.575**	0.463	0.588	0.689
PnP+RANSAC only	NoPoSplat	0.572	0.728	0.833	0.078	0.198	0.394	0.337	0.497	0.646
	Ours (2DGS)	0.588	0.737	0.832	0.085	0.223	0.432	0.344	0.513	0.659
	Ours (2DGS+Align)	0.621	0.760	0.849	0.123	0.279	0.471	0.382	0.540	0.674
	Ours (2DGS+Orient)	0.613	0.756	0.848	0.118	0.267	0.460	0.376	0.537	0.673
	Ours (2DGS+Align+Orient)	0.627	0.766	0.855	0.135	0.289	0.479	0.392	0.547	0.679
w/ Photometric Optimisation	NoPoSplat	0.672	0.791	0.868	0.109	0.256	0.463	0.456	0.593	0.705
	Ours (2DGS)	0.672	0.788	0.859	0.129	0.298	0.515	0.460	0.599	0.713
	Ours (2DGS+Align)	0.686	0.799	0.870	0.136	0.311	0.512	0.474	0.607	0.718
	Ours (2DGS+Orient)	0.679	0.798	0.871	0.141	0.323	0.520	0.475	0.610	0.721
	Ours (2DGS+Align+Orient)	**0.689**	**0.804**	**0.876**	0.156	0.334	0.539	**0.488**	**0.619**	**0.726**

4 Experiments

Datasets and implementation details. Following [2, 3, 51], we train our models on the large-scale RealEstate10K [57] (RE10K) dataset, with the train-test splits used by [51]. RE10K comprises predominantly indoor real-estate videos from YouTube, containing 67,477 training and 7,289 testing scenes, with camera poses computed using COLMAP [34]. For evaluating generalization, we further test on two additional datasets: ACID [30], containing aerial nature scenes captured by drones (with COLMAP-computed poses), and ScanNet [5], an RGB-D indoor scene dataset with distinct camera motion and characteristics. Specifically, we evaluate relative pose and geometry estimation on the ScanNet. Our training broadly follows recent generalizable splatting methods; full details are in supplementary material. Code and models will be released.

4.1 Relative Pose Evaluation

Relative pose is evaluated by computing the AUC of the cumulative pose error curve at three thresholds. We report results deploying a PnP + RANSAC algorithm to align the Gaussian means

Table 2: **Depth estimation for novel views on ScanNet-V1 [5].** We use the novel-view rendered depth accuracy as a holistic measure of 3D reconstruction and interpolation. Our method outperforms all competitors on every metric. Best scores are in **bold**, and top results *without* pose refinement are underlined. Pose-required methods are marked †.

	Pose-required†			*w/o* Pose Refine.					*w* Pose Refine.				
Metric	PixelSplat (3DGS)	MVSplat (3DGS)	DepthSplat (3DGS)	NoPoSplat (3DGS)	Ours (2DGS) $\lambda_a, \lambda_o = 0$	Ours (2DGS) $\lambda_o = 0$	Ours (2DGS) $\lambda_a = 0$	Ours (2DGS) $\lambda_a, \lambda_o \neq 0$	NoPoSplat (3DGS)	Ours (2DGS) $\lambda_a, \lambda_o = 0$	Ours (2DGS) $\lambda_o = 0$	Ours (2DGS) $\lambda_a = 0$	Ours (2DGS) $\boldsymbol{\lambda_a, \lambda_o \neq 0}$
Abs Rel ↓	0.299	0.189	0.135	0.131	0.121	0.109	0.115	0.108	0.126	0.114	0.102	0.106	**0.100**
$\delta_1 < 1.10$ ↑	0.552	0.412	0.578	0.554	0.668	0.679	0.674	0.680	0.567	0.692	0.706	0.704	**0.707**
$\delta_1 < 1.25$ ↑	0.818	0.745	0.864	0.851	0.879	0.890	0.884	0.892	0.861	0.884	0.901	0.898	**0.904**

Table 3: **Depth estimation for source views on ScanNet-V1 [5].** Best self-supervised scores are in **bold**, and top results *without* pose refinement are underlined. Pose-required methods are marked †, and those using extra supervision (e.g., ground-truth depth) are marked ‡ (upper-bound reference).

	Supervised‡	Pose-required†			Pose-free *w/o* Refine.					Pose-free *w* Refine.				
Metric	DUSt3R	pixelSplat (3DGS)	MVSplat (3DGS)	DepthSplat (3DGS)	NoPoSplat (3DGS)	Ours (2DGS) $\lambda_a, \lambda_o = 0$	Ours (2DGS) $\lambda_o = 0$	Ours (2DGS) $\lambda_a = 0$	Ours (2DGS) $\lambda_a, \lambda_o \neq 0$	NoPoSplat (3DGS)	Ours (2DGS) $\lambda_a, \lambda_o = 0$	Ours (2DGS) $\lambda_o = 0$	Ours (2DGS) $\lambda_a = 0$	Ours (2DGS) $\boldsymbol{\lambda_a, \lambda_o \neq 0}$
Abs Rel ↓	0.059	0.288	0.132	0.105	0.121	0.118	0.111	0.114	0.109	0.112	0.105	0.100	0.102	**0.098**
$\delta_1 < 1.10$ ↑	0.886	0.553	0.641	**0.722**	0.662	0.665	0.672	0.671	0.675	0.698	0.705	0.714	0.713	0.716
$\delta_1 < 1.25$ ↑	0.967	0.820	0.891	**0.914**	0.869	0.875	0.886	0.881	0.888	0.883	0.894	0.904	0.900	0.907

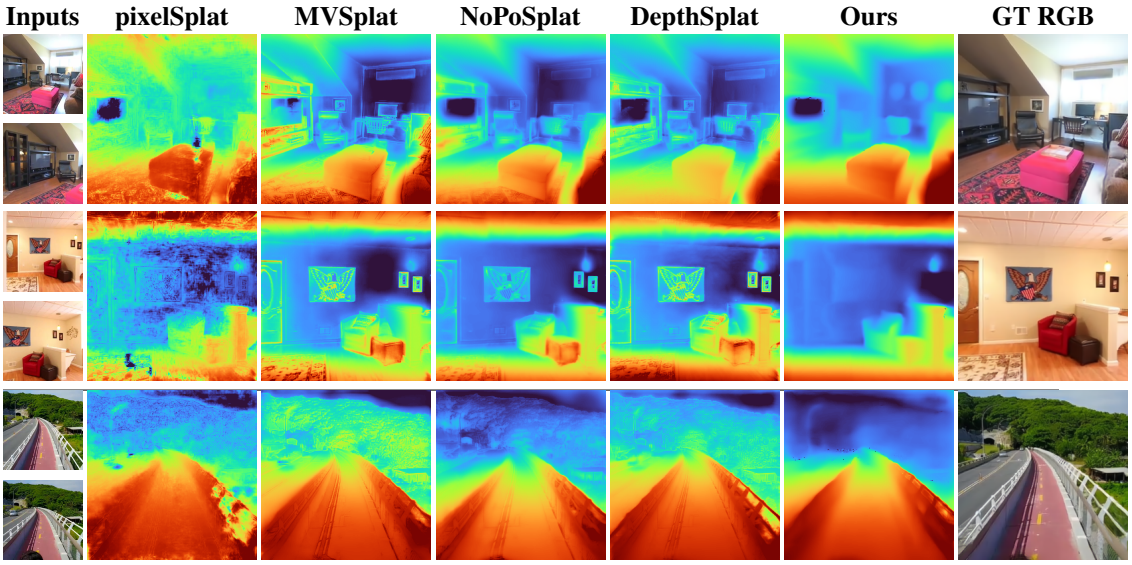

Figure 3: **Qualitative comparison of rendered novel-view depth on RE10k [57] (top two rows) and ACID [30] (bottom row).**

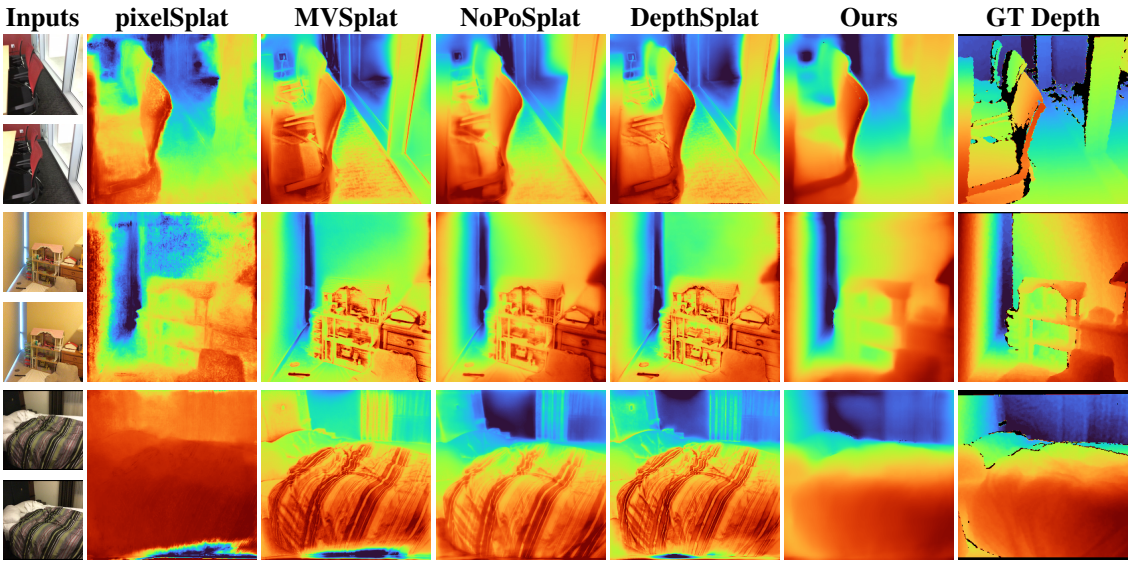

Figure 4: **Qualitative comparison of rendered novel-view depth on ScanNet-V1 [5].**

with image grid as proposed in DUSt3R[42]. NoPosplat [51] proposes a gradient-descent relative pose refinement in which the predicted Gaussians are rendered to generate optimal input image pairs for refining the pose obtained by PnP+RANSAC. Pose Jacobians from [31] are used for this refinement over a fixed number of iterations. Since these Jacobians expect 3D Gaussians, we lift our 2D Gaussians by assigning a small nonzero third scale, to facilitate comparisons.

Table 1 compares relative pose estimation across several methods. CoPoNeRF [17] is trained on RE10K and ACID with explicit pose supervision; DUSt3R [42] uses indoor RGB-D and Internet SfM data (e.g., ScanNet++ [52], MegaDepth [28]) with a 3D regression loss supervising both depth and pose; MASt3R [25] follows DUSt3R's scheme but adds large-scale outdoor sequences (Waymo [36]); RoMA [6] is trained on MegaDepth and ScanNet with depth-and-pose supervision. In contrast, our models use no explicit depth supervision and are trained only on RE10K data. Despite this, we outperform all these methods by a large margin, both on the in-domain RE10K test set and in

zero-shot evaluations on ACID and ScanNet. The sole exception is RoMA [6] on ScanNet-V1, the dataset it was explicitly trained on for relative pose estimation.

Compared to NoPoSplat, our method yields substantial relative-pose gains using only PnP+RANSAC. Incorporating the alignment loss $\mathcal{L}_{align}$ produces marked improvements in both in-domain and cross-domain zero-shot tests, while the orientation loss $\mathcal{L}_{orient}$ provides a further pose-estimation boost, with the combined full loss achieving the best performance. As in NoPoSplat, minimizing the input-image synthesis loss also benefits our pose estimation. Although gains on RE10K and ACID were modest, we observe proportionally larger improvements on ScanNet-V1 with this optimization.

4.2 Geometry Evaluation

Geometric veracity of the estimated 2D/3D Gaussian splats is the key focus of this work. Traditionally, the geometry predicted by feed-forward neural networks is evaluated by measuring the depth errors for the input views. However, input depths do not capture the interpolation capability of predicted Gaussians and are insensitive to the opacity, orientation, and scale. We propose a more holistic evaluation of the predicted scene structure by rendering multiple virtual depth maps from the reconstructed Gaussians and reporting Absolute Relative Error and depth accuracy for two different thresholds. As we do not aim to extrapolate beyond the given view frustum, we use the same view-synthesis test set for depth evaluation. Virtual depth maps are rendered using the ground-truth relative pose w.r.t the first input frame, assuming perfectly aligned multi-view Gaussians. This puts pose-free methods at a severe disadvantage – small pose alignment errors amplify depth errors – yet they outperform pose-aware counterparts by a large margin, as shown in Table 3. Our approach substantially outperforms the NoPoSplat baseline. While the 2DGS parameterization improves the accuracy of structure recovery, the orientation and alignment losses deliver much larger gains. This trend persists even after per-image pose optimization is performed to align the recovered 3D Gaussians isolating pose-estimation error from reconstruction quality.

We compare novel-view depth estimation of our method against pixelSplat [2], MVSplat [3], NoPoSplat [51] and DepthSplat [47] in Figure 3 on RE10K and ACID. MVSplat, DepthSplat and NoPoSplat depths are hypersensitive to texture, while pixelSplat produces notably noisier depths in textureless regions. In contrast, our method yields more plausible depths despite not requiring relative poses. Similar trends are observed in Figure 4 on ScanNet test scenes.

While our primary goal is to predict pixel-aligned, geometrically consistent Gaussians for novel-view depth rendering, we also benchmark source-view depth estimation accuracy of all baseline methods in Table 3. For each method, we report their best depth—whether rendered from Gaussians or predicted by their depth-estimation head—under its best-performing configuration. For example, pixelSplat attains its highest accuracy using rendered depth, whereas MVSplat and DepthSplat perform best with their network-predicted depths. Our method achieves the lowest AbsRel error and performs competitively in thresholded accuracy, slightly trailing DepthSplat [47]. More detailed results for one- and two-view depth prediction are provided in the supplementary material.

4.2.1 Novel View Synthesis Evaluation

While not central to our contributions, we evaluate in-domain novel-view synthesis against relevant baselines. Our method outperforms prior work in novel-view synthesis on Re10K dataset, largely thanks to its warping-free formulation. We also observe improvements over NoPoSplat when training with our proposed loss. Detailed results in the supplementary material.

5 Conclusion

We propose a novel self-supervised, generalizable splatting network that mitigates geometric inconsistencies in Gaussian splat recovery previously overlooked by the community. Our model produces state-of-the-art, geometrically consistent Gaussian splats from just two unposed images. While we train on RE10k using an asymmetric transformer architecture under self-supervision, our core contributions are invariant to these design choices. The priors introduced here will help future work on generalizable splatting and learning-based 3D scene recovery.

References

[1] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *CVPR*, 2018. 4

[2] David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In *CVPR*, 2024. 1, 4, 5, 6, 7, 9

[3] 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*, 2024. 1, 2, 4, 5, 7, 9

[4] Jaeyoung Chung, Jeongtaek Oh, and Kyoung Mu Lee. Depth-regularized optimization for 3d gaussian splatting in few-shot images. In *CVPR*, 2024. 2, 4

[5] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017. 4, 7, 8

[6] Johan Edstedt, Qiyu Sun, Georg Bökman, Mårten Wadenbäck, and Michael Felsberg. Roma: Robust dense feature matching. In *CVPR*, 2024. 4, 8, 9

[7] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE international conference on computer vision*, pages 2650–2658, 2015. 6

[8] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *NeurIPS*, 2014. 4

[9] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *CVPR*, 2018. 4

[10] Yang Fu, Sifei Liu, Amey Kulkarni, Jan Kautz, Alexei A Efros, and Xiaolong Wang. Colmap-free 3d gaussian splatting. In *CVPR*, 2024. 4

[11] Ravi Garg, Vijay Kumar Bg, Gustavo Carneiro, and Ian Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *ECCV*, 2016. 4

[12] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2017. 4

[13] Clement Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth estimation. In *ICCV*, 2019. 4

[14] 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*, 2020. 4

[15] Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. *CVPR*, 2024. 4

[16] Liang Han, Junsheng Zhou, Yu-Shen Liu, and Zhizhong Han. Binocular-guided 3d gaussian splatting with view consistency for sparse view synthesis. *NeurIPS*, 2024. 4

[17] Sunghwan Hong, Jaewoo Jung, Heeseong Shin, Jiaolong Yang, Seungryong Kim, and Chong Luo. Unifying correspondence pose and nerf for generalized pose-free novel view synthesis. In *CVPR*, 2024. 8

[18] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024*, 2024. 3, 4, 5, 6

[19] Jia-Bin Huang, Iain Matthews, and Wolf Kienzle. Deepmvs: Learning multi-view stereopsis. In *CVPR*, 2018. 4

[20] Seunghyun Kang, Hyunwoo Lee, and Hyeongju Chae. Selfsplat: Pose-free and 3d-prior-free generalizable 3d gaussian splatting. *CVPR*, 2025. 4, 5

[21] Nikhil Keetha, Jay Karhade, Krishna Murthy Jatavallabhula, Gengshan Yang, Sebastian Scherer, Deva Ramanan, and Jonathon Luiten. Splatam: Splat, track & map 3d gaussians for dense rgb-d slam. In *CVPR*, 2024. 4

[22] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM TOG*, 2023. 1, 4

[23] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *3DV*, 2016. 4

[24] Jin Han Lee, Youngbok Bae, and In So Kweon Han. Bts: Depth estimation via local planar guidance. *arXiv preprint arXiv:1907.10326*, 2019. 4

[25] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. *arXiv preprint arXiv:2406.09756*, 2024. 1, 4, 8

[26] Jiahe Li, Jiawei Zhang, Xiao Bai, Jin Zheng, Xin Ning, Jun Zhou, and Lin Gu. Dngaussian: Optimizing sparse-view 3d gaussian radiance fields with global-local depth normalization. In *CVPR*, 2024. 4

[27] Mingrui Li, Shuhong Liu, Heng Zhou, Guohao Zhu, Na Cheng, Tianchen Deng, and Hongyu Wang. Sgs-slam: Semantic gaussian splatting for neural dense slam. In *ECCV*, 2024. 4

[28] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *CVPR*, 2018. 8

[29] Zhan Li, Zhang Chen, Zhong Li, and Yi Xu. Spacetime gaussian feature splatting for real-time dynamic view synthesis. In *CVPR*, 2024. 4

[30] 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 *ICCV*, 2021. 7, 8

[31] Hidenobu Matsuki, Riku Murai, Paul HJ Kelly, and Andrew J Davison. Gaussian splatting slam. In *CVPR*, 2024. 4, 8

[32] Zhiyuan Min, Yawei Luo, Jianwen Sun, and Yi Yang. Epipolar-free 3d gaussian splatting for generalizable novel view synthesis. *NeurIPS*, 2024. 4

[33] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *ICCV*, 2021. 4

[34] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 7

[35] 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. 1, 4, 5

[36] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*, 2020. 8

[37] Shengji Tang, Weicai Ye, Peng Ye, Weihao Lin, Yang Zhou, Tao Chen, and Wanli Ouyang. Hisplat: Hierarchical 3d gaussian splatting for generalizable sparse-view reconstruction. *ICLR*, 2025. 4

[38] Benjamin Ummenhofer, Hao Zhou, Jonas Uhrig, Nikolaus Mayer, Eddy Ilg, Alexey Dosovitskiy, and Thomas Brox. Demon: Depth and motion network for learning monocular stereo. In *CVPR*, 2017. 4

[39] Ziyu Wan, Hao Gao, Rui Xiong, and Fang Du. S2gaussian: Sparse-view super-resolution 3d gaussian splatting. In *CVPR*, 2025. 4

[40] Fangjinhua Wang, Silvano Galliani, Christoph Vogel, Pablo Speciale, and Marc Pollefeys. Patchmatchnet: Learned multi-view patchmatch stereo. In *CVPR*, 2021. 4

[41] Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A. Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state. In *CVPR*, 2025. 4

[42] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *CVPR*, 2024. 1, 3, 4, 6, 8

[43] Yunsong Wang, Tianxin Huang, Hanlin Chen, and Gim Hee Lee. Freesplat: Generalizable 3d gaussian splatting towards free view synthesis of indoor scenes. *NeurIPS*, 2024. 1, 4

[44] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *CVPR*, 2024. 4

[45] Haolin Xiong, Sairisheek Muttukuru, Rishi Upadhyay, Pradyumna Chari, and Achuta Kadambi. Sparsegs: Real-time 360° sparse view synthesis using gaussian splatting. In *3DV*, 2025. 4

[46] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, Fisher Yu, Dacheng Tao, and Andreas Geiger. Unifying flow, stereo and depth estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(11):13941–13958, 2023. 1, 4

[47] 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, 4, 5, 9

[48] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *CVPR*, 2024. 4

[49] Zeyu Yang, Hongye Yang, Zijie Pan, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. In *ICLR*, 2024. 4

[50] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *ECCV*, 2018. 4

[51] Botao Ye, Sifei Liu, Haofei Xu, Li Xueting, Marc Pollefeys, Ming-Hsuan Yang, and Peng Songyou. No pose, no problem: Surprisingly simple 3d gaussian splats from sparse unposed images. In *ICLR*, 2025. 1, 2, 3, 4, 5, 6, 7, 8, 9

[52] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *ICCV*, 2023. 8

[53] Huangying Zhan, Ravi Garg, Chamara Saroj Weerasekera, Kejie Li, Harsh Agarwal, and Ian Reid. Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction. In *CVPR*, 2018. 4

[54] Shunyuan Zheng, Boyao Zhou, Ruizhi Shao, Boning Liu, Shengping Zhang, Liqiang Nie, and Yebin Liu. Gps-gaussian: Generalizable pixel-wise 3d gaussian splatting for real-time human novel view synthesis. In *CVPR*, 2024. 1, 4

[55] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3d: A modern library for 3d data processing. *arXiv preprint arXiv:1801.09847*, 2018. 3

[56] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017. 4

[57] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: learning view synthesis using multiplane images. *ACM TOG*, 2018. 2, 3, 7, 8

[58] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *CVPR*, 2022. 4

[59] Zehao Zhu, Zhiwen Fan, Yifan Jiang, and Zhangyang Wang. Fsgs: Real-time few-shot view synthesis using gaussian splatting. In *ECCV*, 2024. 2, 4

[60] Zihan Zhu, Songyou Peng, Viktor Larsson, Zhaopeng Cui, Martin R Oswald, Andreas Geiger, and Marc Pollefeys. Nicer-slam: Neural implicit scene encoding for rgb slam. In *3DV*, 2024. 4

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: See Section 3, Section 4.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See Section 1, Section 4.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: See Section 1, Section 3.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: See Section 4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: See Section 4, and supplemental material. We will release our code and model.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: See Section 4, and supplemental material.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [NA]

Justification: This paper does not include experiments reporting error bars.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See Section 4, and supplemental material.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: This paper conforms, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: See supplemental material.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to

generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have correctly cited and credited assets used by our work.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: This paper does not release new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: LLM is used only for editing and formatting.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.