DN-Splatter: Depth and Normal Priors for Gaussian Splatting and Meshing

Matias Turkulainen*[1] Xuqian Ren*[2] Iaroslav Melekhov[3] Otto Seiskari[4] Esa Rahtu[2]
Juho Kannala[3,4]
[1] ETH Zurich, [2] Tampere University, [3] Aalto University, [4] Spectacular AI

Abstract

High-fidelity 3D reconstruction of common indoor scenes is crucial for VR and AR applications. 3D Gaussian splatting, a novel differentiable rendering technique, has achieved state-of-the-art novel view synthesis results with high rendering speeds and relatively low training times. However, its performance on scenes commonly seen in indoor datasets is poor due to the lack of geometric constraints during optimization. In this work, we explore the use of readily accessible geometric cues to enhance Gaussian splatting optimization in challenging, ill-posed, and textureless scenes. We extend 3D Gaussian splatting with depth and normal cues to tackle challenging indoor datasets and showcase techniques for efficient mesh extraction. Specifically, we regularize the optimization procedure with depth information, enforce local smoothness of nearby Gaussians, and use off-the-shelf monocular networks to achieve better alignment with the true scene geometry. We propose an adaptive depth loss based on the gradient of color images, improving depth estimation and novel view synthesis results over various baselines. Our simple yet effective regularization technique enables direct mesh extraction from the Gaussian representation, yielding more physically accurate reconstructions of indoor scenes. Our code will be released in `https://github.com/maturk/dn-splatter`.

1. Introduction

The demand for high-fidelity 3D reconstruction of typical environments is increasing due to VR and AR applications. However, photorealistic and accurate 3D reconstruction of common indoor scenes from casually captured sensor data remains a persistent challenge in 3D computer vision. Textureless and less-observed regions cause ambiguities in reconstructions and do not provide enough constraints for valid geometric solutions. Recently, neural implicit representations have achieved success in high-fidelity 3D reconstruction by representing scenes as continuous volumes with fully differentiable properties [2, 25, 30, 50]. However, the reconstruction of everyday indoor scenes still poses challenges, even for state-of-the-art methods. These methods rarely achieve good results in both photorealism and geometry reconstruction and often suffer from long training and rendering times, making them inaccessible for general use and VR/AR applications.

3D Gaussian splatting [19] introduces a novel method for rendering by representing a scene by many differentiable 3D Gaussian primitives with optimizable properties. This explicit representation enables real-time rendering of large, complex scenes–a capability that most neural implicit models lack. 3DGS is a more interoperable scene representation compared to neural methods since it encodes a scene's appearance and geometry directly through the location, shape, and color attributes of Gaussians. However, due to the lack of 3D cues and surface constraints during optimization, artifacts and ambiguities are likely to occur, resulting in floaters and poor surface reconstruction. Scenes can often contain millions of Gaussians, and their properties are directly modified by gradient descent based on photometric losses only. Little focus has been given to exploring better regularization techniques that result in visually and geometrically smoother and more plausible 3D reconstructions that can be converted into meshes, an important downstream application.

Although many modern smartphones are equipped with low-resolution depth sensors, these are rarely used for novel-view synthesis tasks. Motivated by this and advances in depth and normal estimation networks [1, 3, 7, 53], we explore the regularization of 3D Gaussian splatting with these geometric priors. Our goal is to enhance both photorealism and surface reconstruction in challenging indoor scenes. By designing an optimization strategy for 3D Gaussian splatting with depth and normal priors, we improve novel-view synthesis results over baselines whilst better respecting the captured scene geometry. We regularize the position of Gaussians with an edge-aware depth constraint and estimate normals from Gaussians to align them with the real surface boundaries estimated via monocular networks. We show how this simple regularization strategy, illustrated in Fig. 1,

*Denotes equal contribution

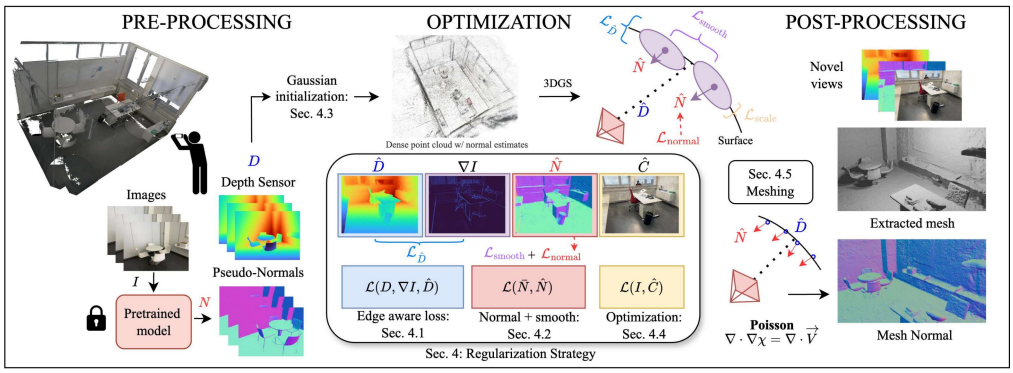


Figure 1. **Overview:** We use depth and normal priors obtained from common handheld devices and general-purpose networks to enhance Gaussian splatting reconstruction quality. By regularizing Gaussian positions, local smoothness, and orientations, we demonstrate improvements in novel-view synthesis and achieve more accurate mesh reconstructions on a variety of challenging indoor room datasets.

enables the extraction of meshes from the Gaussian scene representation, resulting in smoother and more geometrically accurate reconstructions. In summary, we make the following contributions:

- We design an edge-aware depth loss for Gaussian splatting depth regularization to improve reconstruction on indoor scenes with imperfect depth estimates.
- We use monocular normal priors to align Gaussians with the scene and demonstrate how this aids reconstruction.
- We show how regularization with depth and normal cues enables efficient mesh extraction directly from the Gaussian scene with improved novel-view synthesis.

2. Related work

Here we give a brief overview to image-based rendering (IBR) methods for scene reconstruction and an overview of prior methods utilizing geometry cues for regularization.

Traditional IBR. Reconstructing 3D geometry from images is a longstanding challenge in computer vision. Traditional methods like Structure-from-Motion (SfM) [40, 43] and Multi-view Stereo (MVS) [9,60] techniques have focused on reconstructing geometry via a sparse set of multi-view consistent 3D points obtained by triangulation of features from images [21, 39]. Learning-based approaches [13, 27, 29, 31] usually replace parts of the pipeline with learnable modules, leading to improvements in the generalizability of the methods. Other work focus on normal estimation [41] and constructing triangle meshes [18, 26].

Neural implicit IBR. Most success obtaining both photorealisic and geometrically accurate 3D reconstruction has been achieved with neural-based inverse rendering methods, most notably that of NeRF [30], which represents scenes as volumes with attributes encoded within a neural network and applies volume rendering [17] to achieve impressive novel-view results. However, the 3D geometry extracted from NeRFs are often poorly defined and suffer from artifacts and floaters. Subsequent work has focused on improving the rendering quality and scene reconstruction through regularization techniques [6, 38] or by adopting other scene representations such as signed distance functions [25, 50, 59] (SDFs) to improve geometry extraction.

Prior regularization. Prior regularization of neural implicit models has been an active area of research. Previous NeRF-based approaches add depth regularization to explicitly supervise ray termination [6, 38, 51] or impose smoothness constraints [33] on rendered depth maps. Other works explore regularizing with multi-view consistency [6, 10, 24] in sparse view settings. For SDF-based models, Manhattan-SDF [12] uses planar constraints on walls and flat surfaces to improve indoor reconstruction, and MonoSDF [59] uses depth and normal monocular estimates for scene geometry regularization. In this work, we investigate the regularization of 3D Gaussian splatting optimization with depth and normal priors to enhance photometric and geometric reconstruction.

Meshable implicit representations. Surface extraction as triangle meshes is an important problem since most computer graphics pipelines still rely on triangle rasterization. Watertight meshes also provide a good approximation of scene geometry and surface quality, leading to the development of various metrics for mesh quality. Prior work has focused on extracting meshes from NeRF representations [36, 45, 46] with some success, but these methods often rely on expensive post-refinement stages. Most State-of-the-Art techniques use SDF or occupancy representations [25, 50, 54, 59] combined with marching cubes [26] to achieve finer details. These methods involve querying and evaluating dense 3D volumes, often at multiple levels of detail, and are generally slow to train. In this work, we investigate extracting meshable surfaces directly from a Gaussian scene representation.

Meshable 3D Gaussians. Extracting meshable surfaces from Gaussian primitives is a relatively new topic. Keselman *et al.* [20] propose generating an oriented point set from a trained Gaussian scene to be meshed with Poisson

reconstruction [18], using back-projected depth maps and analytically estimated normals. However, without regularization, this approach results in noisy point clouds that are difficult to mesh. NeuSG [4] addresses this by jointly training a dense SDF neural implicit model [50] alongside a Gaussian scene, aligning Gaussians with SDF-estimated normals. The authors improve surface extraction from the Gaussian scene using SDF guidance; however, the approach requires long training times — over 16 hours on high-end GPUs — diminishing the appeal of 3DGS.

SuGaR [11] proposes treating the positions of Gaussians as intersections of a level set and optimizes a signed-distance loss to converge Gaussians to the surface. Normals are estimated from the derivative of the signed distance, similar to Keselman *et al*. However, due to the lack of geometric priors, the reconstructions remain noisy (we show experimental comparing SuGaR meshing strategy in the supplement). SuGaR refines the coarse mesh with additional optimization, making the process computationally costly. In contrast, 2DGS [16] proposes to use a Gaussian surfel representation to explicitly model planar surfaces. However, we show in our experiments that without prior regularization, results on outward-facing indoor reconstructions are poor.

3. Preliminaries

Our work builds on 3D Gaussian splatting (3DGS, [19]) and we briefly describe the rasterization algorithm. 3DGS represents a scene using differentiable 3D Gaussian primitives, parameterized by their mean $\boldsymbol{\mu} \in \mathbb{R}^3$, a covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{3\times3}$ decomposed into a scaling vector $\boldsymbol{s} \in \mathbb{R}^3$ and a rotation quaternion $\boldsymbol{q} \in \mathbb{R}^4$, along with opacity $o \in \mathbb{R}$ and color $\boldsymbol{c} \in \mathbb{R}^3$, represented via spherical harmonics. Rendering a new view involves projecting 3D Gaussians into 2D Gaussians in camera space. These 2D Gaussians are z depth-sorted and alpha-composited using the discrete volume rendering equation to produce a pixel color $\hat{\mathbf{C}}$:

$$\hat{\mathbf{C}} = \sum_{i\in N} \mathbf{c}_i \alpha_i T_i, \text{ where } T_i = \prod_{j=1}^{i-1}(1-\alpha_j) \tag{1}$$

where T_i is the accumulated transmittance at pixel location p and α_i is the blending coefficient for a Gaussian with center μ_i in screen space:

$$\alpha_i = o_i \cdot \exp\left(-\frac{1}{2}(\boldsymbol{p}-\boldsymbol{\mu}_i)^\intercal \boldsymbol{\Sigma}_i^{-1}(\boldsymbol{p}-\boldsymbol{\mu}_i)\right). \tag{2}$$

The scene is typically initialized with sparse SfM points obtained from a pre-processing step [40, 41]. In this work, we also explore initialization using sensor depth readings. The Gaussian scene is optimized using the Adaptive Density Control (ADC) algorithm [19], which progressively culls, splits, and duplicates Gaussians in the scene at fixed intervals based on Gaussian opacity, screen-space size, and the magnitude of the gradient of Gaussian means, respectively.

4. Method

We address the problem of achieving high-fidelity reconstruction of common indoor scenes that is both photorealistic and geometrically precise. In Section 4.1 we utilize sensor and monocular depth priors to regularize Gaussian positions with an edge-aware loss. Next, we extract normal directions from Gaussians and utilize normal cues for regularization in Section 4.2. Additionally, we add a smoothing prior on rendered normal maps to better align nearby Gaussians during optimization in Section 4.4 and initialize the Gaussian scene using dense depth information in Section 4.3. Lastly, in Section 4.5, we use the optimized Gaussian scene to directly extract meshes using Poisson surface reconstruction.

4.1. Leveraging depth cues

Depth prediction. Per-pixel z–depth estimates $\hat{D}$ are rendered using the discrete volume rendering approximation similar to color values:

$$\hat{D} = \sum_{i\in N} d_i \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j) \tag{3}$$

where d_i is the i^{th} Gaussian z-depth coordinate in view space. Since 3DGS does not sort Gaussians individually per-pixel along a viewing ray, and instead relies on a single global sort for efficiency; this is only an approximation of per-pixel depth as explained in [35]. Nevertheless, this approximation remains effective, particularly for more regular geometries typically encountered in indoor datasets. We normalize depth estimates with the final accumulated transmittance T_i per pixel, ensuring that we correctly estimate a depth value for pixels where the accumulated transmittance does not equal 1. We rasterize color and depths simultaneously per pixel in a single CUDA kernel forward pass, improving inference and training speed compared to separate rendering steps.

Sensor depth regularization. We directly apply depth regularization on predicted depth maps for datasets containing lidar or sensor depth measurements [37, 44, 57]. Common commercial depth sensors, especially low-resolution variants found in consumer devices like iPhones, often produce non-smooth edges at object boundaries and provide inaccurate readings. Based on this observation and inspired by [5, 22], we propose a gradient-aware depth loss for adaptive depth regularization based on the current RGB image. The depth loss is lowered in regions with large image gradients, signifying edges, ensuring that regularization is more enforced on smoother texture-less regions that typically pose challenges for photometric regularization alone. Additionally, our experiments (*cf*. Table 6) show that using a logarithmic penalty results in smoother reconstructions compared to linear or quadratic penalties. This insight drives our formulation of the gradient-aware depth loss, which effectively balances the regularization across different regions of the image, adapting

to the scene's geometry and texture complexity. We define the gradient-aware depth loss as follows:

$$\mathcal{L}_{\hat{D}} = g_{\text{rgb}} \frac{1}{|\hat{D}|} \sum \log(1 + \|\hat{D} - D\|_1) \tag{4}$$

where $g_{\text{rgb}} = \exp(-\nabla I)$ and ∇I is the gradient of the current aligned RGB image. $|\hat{D}|$ indicates the total number of pixels in $\hat{D}$.

Monocular depth regularization. For datasets containing no depth data, we rely on scale-aligned monocular depth estimation networks for regularization. We use off-the-shelf monocular depth networks, such as ZoeDepth [3] and DepthAnything [53], for dense per-pixel depth priors. We address the scale ambiguity between estimated depths and the scene by comparing them with sparse SfM points, similar to prior work [5, 59]. Specifically, for each monocular depth estimate D_{mono}, we align the scale to match that of the sparse depth map D_{sparse} obtained by projecting SfM points to the camera view. We solve for a per-image scale a and shift b parameter using the closed-form linear regression solution to:

$$\hat{a}, \hat{b} = \arg\min_{a,b} \sum_{ij} \|(a * D_{\text{mono},ij} + b) - D_{\text{sparse},ij}\|_2^2, \tag{5}$$

where we denote $D_{\text{sparse},ij}$ and $D_{\text{mono},ij}$ as per-pixel correspondences between the two depth maps. We then apply the same loss as in Eq. (4) for regularization.

4.2. Leveraging normal cues

Normal prediction. During optimization, we expect Gaussians to become flat, disc-like, with one scaling axis much smaller than the other two. This smaller scaling axis serves as an approximation of the normal direction. Specifically, we define a geometric normal for a Gaussian using a rotation matrix $\mathbb{R} \in SO(3)$, obtained from its quaternion $\boldsymbol{q}$, and scaling coefficients $\boldsymbol{s} \in \mathbb{R}^3$:

$$\hat{\boldsymbol{n}}_i = R \cdot \text{OneHot}(\arg\min(s_1, s_2, s_3)), \tag{6}$$

where $\text{OneHot}(.) \in \mathbb{R}^3$ returns a unit vector with zeros everywhere except at the position where the scaling $\boldsymbol{s}_i = (s_1, s_2, s_3)$ is minimum. R is obtained from the quaternion $q = (w, x, y, z)^\top$ using:

$$R = \begin{bmatrix} 1 - 2\left(y^2 + z^2\right) & 2\left(xy - wz\right) & 2\left(xz + wy\right) \\ 2\left(xy + wz\right) & 1 - 2\left(x^2 + z^2\right) & 2\left(yz - wx\right) \\ 2\left(xz - wy\right) & 2\left(yz + wx\right) & 1 - 2\left(x^2 + y^2\right) \end{bmatrix} \tag{7}$$

We minimize one of the scaling axes during training to force Gaussians to become disc-like surfels:

$$\mathcal{L}_{\text{scale}} = \sum_i \| \arg\min(\boldsymbol{s}_i) \|_1. \tag{8}$$

To ensure correct orientations, we flip the direction of the normals at the beginning of training if the dot product between the current camera viewing direction and the Gaussian normal is negative. Normals are transformed into camera space using the current camera transform and alpha-composited according to the rendering equation to provide a single per-pixel normal estimate:

$$\hat{\boldsymbol{N}} = \sum_{i \in N} \hat{\boldsymbol{n}}_i \alpha_i T_i. \tag{9}$$

This approach derives normal estimates directly from the geometry of Gaussians. Consequently, during back-propagation, adjustments to scale and rotation parameters, *i.e.*, covariance matrices, directly lead to updates in normal estimates. Therefore, no additional learnable parameters are needed. Intuitively, this results in Gaussians better conforming to the scene's geometry, as their orientations and scales are compelled to align with the surface normal.

Monocular normals regularization. Gao *et al.* [8] propose using pseudo-ground truth normal maps estimated from the gradient of rendered depths for supervision, referred to as $\nabla \hat{D}$. However, due to noise in rendered depth maps, especially in complex scenes, this method results in artifacts. Instead, we supervise predicted normals using monocular cues obtained from Omnidata [7], which provide much smoother normal estimates. Fig. 2 highlights this difference. We regularize with an L1 loss:

$$\mathcal{L}_{\hat{N}} = \frac{1}{|\hat{N}|} \sum \|\hat{\boldsymbol{N}} - \boldsymbol{N}\|_1. \tag{10}$$

We further apply a prior on the total variation of predicted normals, encouraging smooth normal predictions at neighboring pixels with:

$$\mathcal{L}_{\text{smooth}} = \sum_{i,j} \left(|\nabla_i \hat{\boldsymbol{N}}_{i,j}| + |\nabla_j \hat{\boldsymbol{N}}_{i,j}| \right), \tag{11}$$

where $\hat{N}_{ij}$ represents estimated normal values at pixel position (i, j) and ∇ represents the finite difference operator that convolves its input with $[-1, 1]$ for the i-axis and $[-1, 1]^\top$ for the j-axis. Thus, our normal regularization loss is defined as $\mathcal{L}_{\text{normal}} = \mathcal{L}_{\hat{N}} + \mathcal{L}_{\text{smooth}}$.

4.3. Gaussian initialization

Rather than relying on SfM points for initialization, we make use of sensor depth readings, where available, by back-projecting depths into world coordinates for dense scene initialization. Additionally, we initialize Gaussian orientations by estimating normal directions from the initial point cloud using [61], aligning the Gaussian orientations q with these estimated normals using Eq. (7), and setting one of the scaling axes smaller than the others. This initialization helps normal estimation convergence.

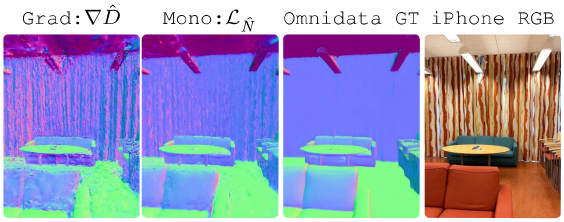


Figure 2. **Depth gradient vs. monocular normal supervision strategy.** (a) We observe that using pseudo normal maps derived from the gradient of rendered depths [8] for supervision leads to noisy predicted normals compared to (b) normal supervision by estimates from a pretrained (c) Omnidata model [7].

4.4. Optimization

The final loss we use for optimization is defined as follows:

$$\mathcal{L} = \mathcal{L}_{\hat{C}} + \lambda_{\text{d}}\mathcal{L}_{\hat{D}} + \mathcal{L}_{\text{scale}} + \underbrace{(\lambda_{\text{n}}\mathcal{L}_{\hat{N}} + \lambda_{\text{s}}\mathcal{L}_{\text{smooth}})}_{\mathcal{L}_{\text{normal}}}, \quad (12)$$

where $\mathcal{L}_{\hat{C}}$ is the original photometric loss proposed in [19]. We set $\lambda_{\text{d}} = 0.2$, $\lambda_{\text{n}} = 0.1$, and $\lambda_{\text{s}} = 0.1$ in our experiments.

4.5. Meshing

After optimizing with our depth and normal regularization using Eq. (12), we apply Poisson surface reconstruction [18] to extract a mesh. With our regularization strategy, we ensure that the positions of Gaussians are well-distributed and aligned along the surface of the scene. We directly back-project rendered depth and normal maps from training views to create an oriented point set for meshing. Qualitative and quantitative differences between various Poisson methods and comparisons with the meshing approach proposed in SuGaR [11] are given in the Supplementary material.

5. Experiments

In this section, we demonstrate the proposed regularization strategy on mesh extraction and novel-view synthesis results using indoor datasets. We also give insight regarding various depth supervision strategies and initialization schemes enabled by sensor depth data.

Datasets. We focus on indoor datasets and consider the following: a) MuSHRoom [37]: a real-world indoor dataset containing separate training and evaluation camera trajectories; and b) ScanNet++ [57]: a real-world indoor dataset with high fidelity 3D geometry and RGB data.

Baselines. We consider a range of baselines including implicit NeRF and SDF based representations and explicit Gaussian based methods. We consider a) state-of-the-art NeRF-based method Nerfacto [45]; b) its depth regularized version Depth-Nerfacto with a direct loss on ray termination distribution for depth supervision similar to DS-NeRF [6]; c) Neusfacto [58] and MonoSDF [59] for SDF-based implicit surface reconstruction; d) baseline 3DGS Splatfacto method based on Nerfstudio `v1.1.3` [45]; e) SuGaR [11] – a 3DGS variant for mesh reconstruction; and f) the recent 2DGS [16] method. In addition, for mesh reconstruction we consider g) traditional Poisson meshing utilizing back-projected sensor depth readings.

Evaluation metrics. We follow standard practice and report PSNR, SSIM and LPIPS metrics for color images and common depth metrics, similar to [23, 28, 32, 42, 47, 51, 62], to analyze depth quality for datasets containing ground truth sensor data. For mesh evaluation, we follow evaluation protocols from [37,49] and report Accuracy (Acc.), Completion (Comp.), Chamfer-L_1 distance (C-L_1), Normal Consistency (NC), and F-scores (F1) with a threshold of 5cm.

Implementation details. The proposed method is implemented in PyTorch [34] and `gsplat` [55] (`v1.0.0`). We train all models for 30k iterations. To obtain monocular normal cues, we propagate RGB images through the pre-trained Omnidata model [7]. We initialize the Gaussian scene using 1M back-projected points from training dataset sensor depths. For Poisson reconstruction, we extract a total of 2 million points and use a depth level of 9 for all methods, unless otherwise stated. All meshes are extracted using back-projection of depth and normal maps besides Neusfacto and MonoSDF (marching cubes) and 2DGS (TSDF). More settings can be seen in the supplementary material.

Gaussian initialization. For all Gaussian based baselines (Splatfacto, SuGaR, 2DGS, and DN-Splatter) we utilize 1M back-projected sensor depth points for initialization of the Gaussian scene unless otherwise stated. We compare sparse COLMAP [40, 41] SfM initialization and sensor depth initialization strategies in Table 5.

5.1. Mesh evaluation

We demonstrate the effectiveness of the proposed regularization strategy on scene geometry by extracting meshes directly after optimization, without any additional refinement steps. In Table 1 and Table 2 we show how incorporating geometric cues enables competitive mesh extraction on scenes from the MuSHRoom [37] and ScanNet++ [57] datasets. On challenging indoor scenes, neither NeRF [45] nor Gaussian-based methods [11, 16] perform well for indoor reconstruction. Improving depth and normal estimation over the baseline Splatfacto [45] makes our method competitive even against the more computationally expensive SDF approaches [58, 59]. NeRF-based methods fail to consistently achieve similar results, with depth supervision sometimes hindering mesh performance. Fig. 3 provides a qualitative comparison of our mesh extraction approach with other baselines on the MuSHRoom and ScanNet++ datasets.

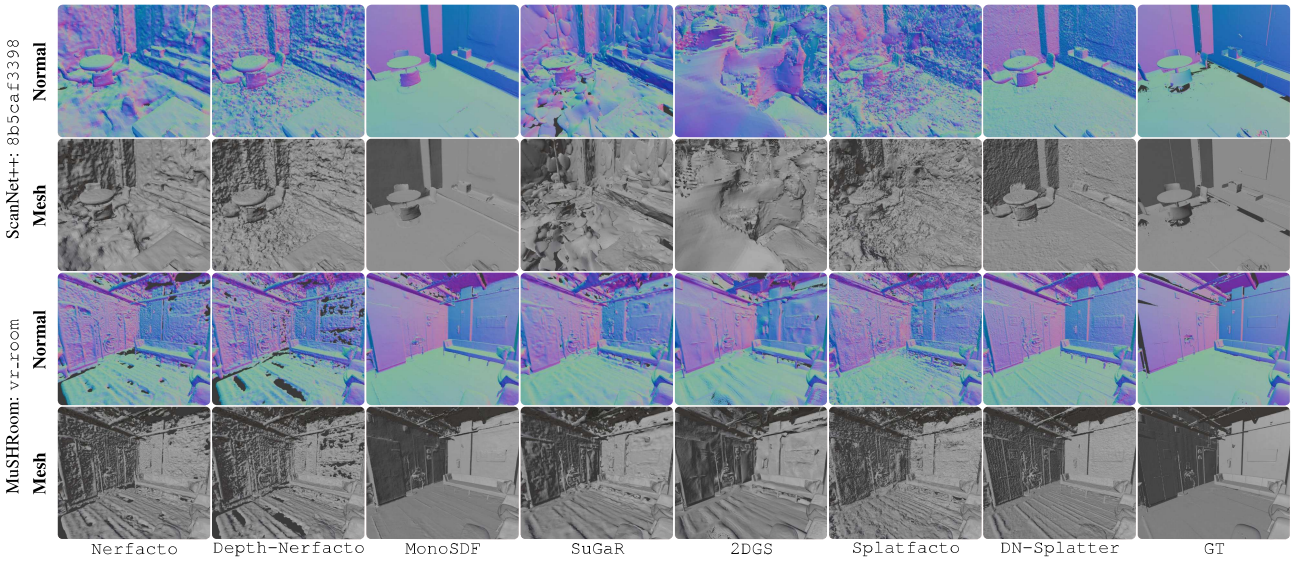


Figure 3. **Mesh reconstruction results**. NeRF variants, even with depth supervision, suffer from artefacts and floaters in reconstruction. The Gaussian based methods Splatfacto, SuGaR, and 2DGS are trained on only photometric losses and thus severly struggle to capture the scene geometry in low texture environments. However, adding depth and normal supervision with DN-Splatter greatly aids reconstruction quality.

		Sensor Depth Loss	Algorithm	Accuracy ↓	Completion ↓	Chamfer-L_1 ↓	Normal Consistency ↑	F-score ↑	Num GS
	Traditional Poisson	–	Poisson	.0399	.0222	.0306	.8688	.8823	-
NeRF	Nerfacto [45]	–	Poisson	.0430	.0578	.0504	.7822	.7212	-
NeRF	Depth-Nerfacto [45]	✓	Poisson	.0447	.0557	.0502	.7614	.6966	-
SDF	MonoSDF [59]	✓	Marching-Cubes	<u>.0310</u>	**.0190**	<u>.0250</u>	**.8846**	<u>.9211</u>	-
Gaussian	SuGaR [11]	–	Poisson+IBR	.0656	.0583	.0620	.8031	.6378	700K
Gaussian	2DGS [16]	–	TSDF	.0731	.0642	.0687	.8008	.6039	2.6M
Gaussian	Splatfacto [45]	–	Poisson	.0749	.0555	.0652	.7727	.5835	1.18M
Gaussian	DN-Splatter (Ours)	✓	Poisson	**.0239**	<u>.0194</u>	**.0216**	<u>.8822</u>	**.9243**	1.18M

Table 1. **Mesh evaluation: MuSHRoom**. Adding depth and normal priors to 3DGS optimization greatly enhances mesh reconstruction on challenging real-world indoor datasets. We report the number of Gaussians in the final optimized scene for Gaussian-based models, with results averaged over 6 scenes: 'coffee_room', 'honka', 'kokko', 'sauna', 'computer', and 'vr_room'. The best and second-best results are indicated with **bold** and <u>underline</u>.

		Sensor Depth Loss	Algorithm	Accuracy ↓	Completion ↓	Chamfer-L_1 ↓	Normal Consistency ↑	F-score ↑	Time (min)
	Traditional Poisson	–	Poisson	<u>.0593</u>	.0574	<u>.0564</u>	.4410	<u>.7923</u>	2.0
NeRF	Nerfacto [45]	–	Poisson	.1305	.1484	.1394	.7153	.4698	8.0
NeRF	Depth-Nerfacto [45]	✓	Poisson	.0731	.1647	.1189	.6848	.5018	8.1
SDF	Neusfacto [58]	–	Marching-Cubes	.0736	.1945	.1340	.7159	.4605	40.0
SDF	MonoSDF [59]	✓	Marching-Cubes	**.0303**	<u>.0573</u>	**.0438**	**.8881**	**.8577**	47.5
Gaussian	SuGaR [11]	–	Poisson + IBR	.0940	.1011	.0975	.7241	.4367	70.0
Gaussian	2DGS [16]	–	TSDF	.1272	.0798	.1035	.7799	.4196	33.5
Gaussian	Splatfacto [45]	–	Poisson	.1934	.1503	.1719	.6741	.1790	8.9
Gaussian	DN-Splatter (Ours)	✓	Poisson	.0940	**.0395**	.0667	**.8316**	.7658	36.9

Table 2. **Mesh evaluation: ScanNet++**. The results are averaged over the 'b20a261fdf' and '8b5caf3398' scenes. The best and second best results are marked with **bold** and <u>underline</u>. Training time is reported using a Nvidia 4090 GPU.

5.2. Novel-view synthesis and depth estimation

We show an extensive study on depth and normal supervision and its effect on novel-view synthesis and depth metrics on challenging real-world scenes from the MuSHRoom and ScanNet++ datasets. Table 3 demonstrates that incorporating sensor depth supervision into 3DGS enhances both depth and RGB metrics compared to scenarios without geometric supervision. We qualitatively highlight the improvements in novel-view synthesis and depth quality, including the reduction of floaters and artifacts in Fig. 4.

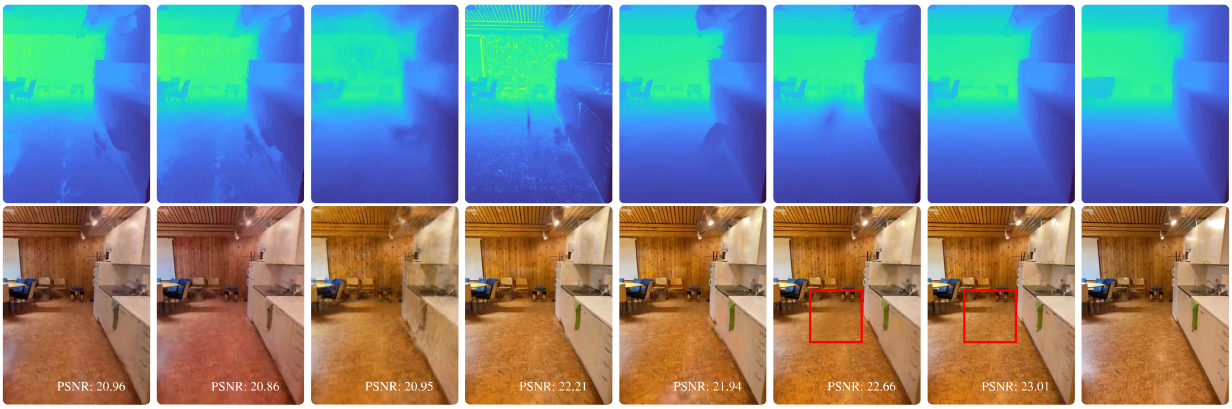


Figure 4. **Qualitative comparison of depth and RGB renders against a variety of baselines.** DN-Splatter achieves the highest novel view synthesis results compared to NeRF, SDF, and Gaussian based methods.

		Sensor Depth Loss	Abs Rel ↓	Sq Rel ↓	RMSE ↓	$\delta < 1.25$ ↑	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF	Nerfacto [45]	–	.0862 / .0747	.0293 / .0141	.0794 / .0667	.9335 / .9428	20.86 / 20.66	.7859 / .7633	.2321 / .2702
	Depth-Nerfacto [45]	✓	.0727 / .0563	.0155 / .0283	**.0583** / .2840	.9469 / .9389	21.24 / 18.93	.7832 / .7023	.2414 / .3978
SDF	MonoSDF [59]	✓	.0555 / .0911	.0263 / .0804	.2493 / .4535	.9353 / .8825	20.68 / 20.16	.7357 / .7653	.3590 / .2261
Explicit	SuGaR [11]	–	.1213 / .1174	.1258 / .1474	.5562 / .5820	.8412 / .8564	20.52 / 18.18	.7740 / .7125	.2427 / .2959
	2DGS [16]	–	.0864 / .0923	.0612 / .0583	.3799 / .3132	.8820 / .8927	22.52 / 21.73	.8185 / .7898	.1773 / .1911
	Splatfacto [45]	–	.0787 / .0817	.0364 / .0521	.2407 / .2941	.9072 / .9061	24.44 / 21.33	.8486 / .7821	.1387 / .2240
	Splatfacto + $\mathcal{L}_{\hat{D}}$	✓	.0234 / .0364	.0092 / .0145	.1293 / .1486	.9849 / **.9745**	**24.77** / 21.95	.8538 / .7948	**.1238** / .1852
	Splatfacto + $\mathcal{L}_{\hat{D}}$ + $\mathcal{L}_{\text{normal}}$	✓	.0241 / **.0340**	.0094 / **.0123**	.1308 / **.1472**	.9848 / **.9777**	24.67 / **21.99**	.8517 / .7941	.1275 / .1864
	DN-Splatter (Ours)	✓	**.0228** / .0354	**.0089** / .0214	.1280 / .2032	**.9854** / .9683	24.58 / 21.89	**.8558 / .7984**	.1293 / **.1797**

Table 3. **Depth estimation and novel view synthesis: MuSHRoom**. The reported results are reported for two distinct evaluation datasets: `left/right` where `left` is a test set obtained by sampling uniformly every 10 frames within the training sequence and `right` is a test split obtained from a different camera trajectory with no overlap with the training sequence. Results are averaged over 6 scenes.

Method	C-L_1 ↓	NC ↑	F1↑	PSNR ↑	SSIM ↑	Time (min)
Splatfacto ($\mathcal{L}_{\text{rgb}}$) [19, 45]	7.0	78.0	56.3	26.6	.870	12.9
+ Normal ($\mathcal{L}_{\hat{N}}$)	7.7	81.3	60.2	26.7	.874	17.2
+ Normal ($\mathcal{L}_{\hat{N}}$ + $\mathcal{L}_{\text{smooth}}$)	6.7	82.6	58.9	26.4	.866	17.3
+ Depth ($\mathcal{L}_{\hat{D}}$)	2.6	86.8	86.6	**27.1**	**.885**	13.1
+ Both ($\mathcal{L}_{\hat{D}}$ + $\mathcal{L}_{\hat{N}}$)	**2.4**	**90.0**	**87.7**	26.9	.883	17.5
+ Both ($\mathcal{L}_{\hat{D}}$ + $\mathcal{L}_{\hat{N}}$ + $\mathcal{L}_{\text{smooth}}$)	2.5	89.6	87.1	26.8	.879	17.9

Table 4. **Comparison of geometric supervision strategies**. Geometric cues significantly improve reconstruction quality. We report mesh, novel view, and training time metrics (Nvidia 4090 GPU, 30k iterations) on the 'VR room' sequence from MuSHRoom

Initialization Method	# Init GS	C-L_1 ↓	NC ↑	F1↑	PSNR ↑	SSIM ↑
COLMAP SfM [41]	∼54.8K	.0242	.8713	.9066	24.23	.8411
Sensor Depth	50 K	.0242	.8725	.9074	24.16	.8389
Sensor Depth	100 K	.0241	.8714	.9079	24.19	.8395
Sensor Depth	500 K	.0239	.8730	.9091	24.20	.8399
Sensor Depth	1 M	.0215	.8390	.9381	24.58	.8558
Sensor Depth	1.5 M	.0238	.8730	.9171	24.31	.8491

Table 5. **Comparison of SfM and sensor depth initialization.** We compare Gaussian scene initialization using COLMAP [41] SfM points and those obtained from back-projecting training dataset sensor depth readings on the MuSHRoom dataset. We note that sensor depth initialization can improve overall reconstruction results compared to SfM initialization. We use 1M points in our experiments. Results are averaged over 6 scenes.

5.3. Ablation studies

Proposed regularization strategy. We evaluate various design choices of our method in Table 4. The normal loss $\mathcal{L}_{\hat{N}}$ (Eq. (10)) helps align Gaussians along the scene geometry improving the resulting mesh normal-completeness and F-scores. The depth loss $\mathcal{L}_{\hat{D}}$ (Eq. (4)) significantly improves reconstruction quality and novel-view synthesis in ambiguous, textureless regions which are common in indoor scenes. The normal smoothing prior $\mathcal{L}_{\text{smooth}}$ (Eq. (11)) further helps normal-completeness for Poisson meshing with a minimal impact on other metrics. Although the smoothing prior's effect on quantitative metrics is minimal, its importance is evident in the qualitative renders illustrated in Fig. 6.

SfM vs. sensor depth initialization. We compare initialization strategies using sparse COLMAP [40, 41] SfM points and those obtained from back-projecting sensor depth readings. We note that using dense sensor depth data for initialization enhances both mesh and novel-view synthesis metrics, demonstrating the value of incorporating available sensor depth data.

2DGS vs. DN-Splatter. We evaluate our method against

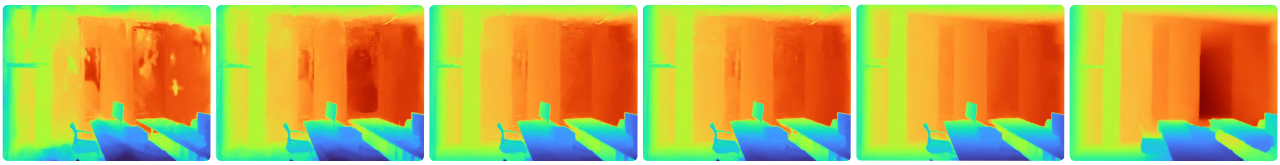


Figure 5. **Qualitative comparison of depth losses: ScanNet++.** We observe that the proposed gradient aware $\mathcal{L}_{\hat{D}}$ regularizer obtains the best qualitative results, mitigating uncertainties at edges from the raw iPhone depth captures. Zoom in to see the details.

Method	Abs Rel ↓	Sq Rel ↓	RMSE ↓	$\delta < 1.25$ ↑	PSNR ↑	SSIM ↑
Splatfacto [45]	.1481	.1345	.5122	.7602	23.41	.9111
Splatfacto + $\mathcal{L}_{\text{MSE}}$	.0551	.0180	.2104	.9647	**23.84**	**.9140**
Splatfacto + $\mathcal{L}_1$	.0351	.0170	.1889	.9758	23.74	.9138
Splatfacto + $\mathcal{L}_{\text{LogL1}}$	.0364	.0180	.1955	.9744	23.77	.9139
Splatfacto + $\mathcal{L}_{\hat{D}}$	**.0285**	**.0162**	**.1790**	**.9782**	23.61	.9122

Table 6. **Quantitative comparison of depth losses: ScanNet++.** We compare various depth supervision strategies and the resulting novel-view and mesh reconstruction results. See Fig. 5 for qualitative comparisons of the losses.

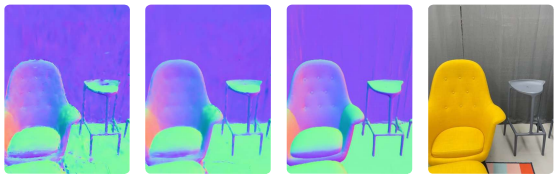


Figure 6. **Qualitative comparison of normal supervision.** We observe that using a direct normal loss $\mathcal{L}_{\hat{N}}$ (a) results in non-smooth surfaces whereas using a normal smoothing prior $\mathcal{L}_{\text{smooth}}$ (b) significantly improves normal estimates according to (c) Omnidata [7] predictions.

Method	C-L_1 ↓	NC ↑	F1↑	PSNR ↑	SSIM ↑
2DGS	.0687	.8008	.6039	22.52	.8185
2DGS w/ Monodepth [3]	.0795	.8558	.5446	21.85	.8009
2DGS w/ Sensor Depth	.0275	**.8830**	.8886	23.02	.8250
Splatfacto	.0652	.7727	.5835	24.44	.8486
Ours w/ Monodepth [3]	.0477	.0860	.6963	24.48	.8509
Ours w/ Sensor Depth	**.0216**	.8822	**.9243**	**24.58**	**.8558**

Table 7. **2DGS vs. DN-Splatter**. We implement depth supervision to the recent 2DGS [16] method and compare with monocular and sensor depth regularization on MuSHRoom. We use the same loss and initialization strategy for fair comparison. Results are averaged over 6 scenes. NVS is reported within the train sequence.

a variant of 2DGS [16] with the same depth regularization strategy in Table 7. Our comparison includes both monocular depth supervision with the Pearson correlation loss [52] and sensor depth supervision. We observe that 2DGS also faces difficulties with textureless regions, which are common in indoor datasets, leading to challenges with planar surfaces.

Depth supervision and losses. We assess the impact of the proposed depth loss Eq. (4) on ScanNet++, with qualitative renders shown in Fig. 5 and quantitative results in Table 6. We compare common depth losses: $\mathcal{L}_{\text{MSE}}$, $\mathcal{L}_1$, $\mathcal{L}_{\text{LogL1}}$ [14], and our edge-aware $\mathcal{L}_{\hat{D}}$. Results indicate that $\mathcal{L}_1$ and $\mathcal{L}_{\text{LogL1}}$ generally yield the best color metrics, with the logarithmic

Method	Acc. ↓	Comp. ↓	C-L_1 ↓	NC ↑	F-score↑
No supervision	.2627	.2091	.2359	.6511	.1343
Monodepth: Zoe-Depth [3]	.1751	.2084	.1918	.7420	.1455
Monodepth: Metric3D [15]	.1798	.2079	.1938	.7358	.1439
Multi-view Stereo (MVSNet) [60]	.3120	.2375	.2748	.6408	.1903
Pearson Loss [52] w/ Metric3D	.1183	.1766	.1474	.7975	.2236
Sensor Depth (iPhone)	**.0609**	**.1433**	**.1021**	**.8130**	**.5833**

Table 8. **Alternative depth regularization strategies**. We compare regularization with monocular [3, 15] and Multi-view Stereo depth estimates [15] using our $\mathcal{L}_{\hat{D}}$ loss, the Pearson correlation loss (patch-based) [52] for relative depth supervision with Zoe-Depth estimates, as well as utilizing iPhone sensor depth supervision on the 'b20a261fdf' scene from ScanNet++.

variant providing the smoothest reconstructions. Further details and comparisons with ground truth Faro lidar scans are available in the supplementary material.

Lastly, Table 8 evaluates our regularization strategy against common alternatives. We compare reconstruction with SotA monocular [3, 15] and multi-view [60] networks using the alignment strategy outlined in Section 4.1 using Eq. (5). We also examine the patch-based Pearson correlation loss [52] for relative depth supervision. While it performs better than naive monocular depth supervision, reconstruction quality is still inferior to using iPhone depths. Despite their low resolution and inaccuracies (mainly at object edges), sensor depths remain practical for real-world indoor scenes. Further research is needed to improve monocular depth supervision performance.

6. Conclusion

We presented DN-Splatter, a method for depth and normal regularization of 3DGS to address photorealistic and geometrically accurate reconstruction of challenging indoor datasets. This simple yet effective strategy enhances novel-view metrics and significantly improves the surface quality extracted from a Gaussian scene. We demonstrated that prior regularization is essential for achieving more geometrically valid and consistent reconstructions in challenging indoor scenes.

7. Acknowledgments

We thank Tobias Fischer, Songyou Peng, and Philipp Lindenberger for their fruitful discussions. We thank Jiepeng

Wang and Marcus Klasson for their help in proof reading. We acknowledge funding from the Academy of Finland (grant No. 327911, 353138, 324346, and 353139) and support from the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Allice Wallenberg Foundation. MT was funded by the Finnish Center for Artificial Intelligence (FCAI).

References

[1] Gwangbin Bae and Andrew J. Davison. Rethinking inductive biases for surface normal estimation. In *CVPR*, 2024. 1

[2] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *CVPR*, 2022. 1

[3] Shariq Farooq Bhat, Reiner Birkl, Diana Wofk, Peter Wonka, and Matthias Müller. Zoedepth: Zero-shot transfer by combining relative and metric depth, 2023. 1, 4, 8

[4] Hanlin Chen, Chen Li, and Gim Hee Lee. Neusg: Neural implicit surface reconstruction with 3d gaussian splatting guidance. *arXiv preprint arXiv:2312.00846*, 2023. 3

[5] Jaeyoung Chung, Jeongtaek Oh, and Kyoung Mu Lee. Depth-regularized optimization for 3d gaussian splatting in few-shot images. *arXiv preprint arXiv:2311.13398*, 2023. 3, 4

[6] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised NeRF: Fewer views and faster training for free. In *CVPR*, 2022. 2, 5, 1

[7] Ainaz Eftekhar, Alexander Sax, Jitendra Malik, and Amir Zamir. Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans. In *ICCV*, pages 10786–10796, 2021. 1, 4, 5, 8

[8] Jian Gao, Chun Gu, Youtian Lin, Hao Zhu, Xun Cao, Li Zhang, and Yao Yao. Relightable 3d gaussian: Real-time point cloud relighting with brdf decomposition and ray tracing. *arXiv preprint arXiv:2311.16043*, 2023. 4, 5

[9] Michael Goesele, Noah Snavely, Brian Curless, Hugues Hoppe, and Steven M. Seitz. Multi-view stereo for community photo collections. In *ICCV*, pages 1–8, 2007. 2

[10] Guangcong, Zhaoxi Chen, Chen Change Loy, and Ziwei Liu. Sparsenerf: Distilling depth ranking for few-shot novel view synthesis. *ICCV*, 2023. 2

[11] Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. *CVPR*, 2024. 3, 5, 6, 7, 1, 2

[12] Haoyu Guo, Sida Peng, Haotong Lin, Qianqian Wang, Guofeng Zhang, Hujun Bao, and Xiaowei Zhou. Neural 3d scene reconstruction with the manhattan-world assumption. In *CVPR*, 2022. 2

[13] Wilfried Hartmann, Silvano Galliani, Michal Havlena, Luc Van Gool, and Konrad Schindler. Learned multi-patch similarity. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017. 2

[14] Junjie Hu, Mete Ozay, Yan Zhang, and Takayuki Okatani. Revisiting single image depth estimation: Toward higher resolution maps with accurate object boundaries. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018. 8

[15] Mu Hu, Wei Yin, Chi Zhang, Zhipeng Cai, Xiaoxiao Long, Hao Chen, Kaixuan Wang, Gang Yu, Chunhua Shen, and Shaojie Shen. Metric3d v2: A versatile monocular geometric foundation model for zero-shot metric depth and surface normal estimation. *arXiv preprint arXiv:2404.15506*, 2024. 8

[16] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *SIGGRAPH 2024 Conference Papers*. Association for Computing Machinery, 2024. 3, 5, 6, 7, 8, 1

[17] James T. Kajiya. The rendering equation. In *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '86, page 143–150, New York, NY, USA, 1986. Association for Computing Machinery. 2

[18] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson Surface Reconstruction. In Alla Sheffer and Konrad Polthier, editors, *Symposium on Geometry Processing*. The Eurographics Association, 2006. 2, 3, 5

[19] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM TOG*, 42(4), 2023. 1, 3, 5, 7, 4, 6, 8, 9

[20] Leonid Keselman and Martial Hebert. Flexible techniques for differentiable rendering with 3d gaussians. In *ICCV*, 2023. 2

[21] Georgios Kopanas, Julien Philip, Thomas Leimkühler, and George Drettakis. Point-based neural rendering with per-view optimization. *Computer Graphics Forum (Proceedings of the Eurographics Symposium on Rendering)*, 40(4), June 2021. 2

[22] Elena Kosheleva, Sunil Jaiswal, Faranak Shamsafar, Noshaba Cheema, Klaus Illgner-Fehns, and Philipp Slusallek. Edge-aware consistent stereo video depth estimation. *arXiv preprint arXiv:2305.02645*, 2023. 3

[23] Uday Kusupati, Shuo Cheng, Rui Chen, and Hao Su. Normal assisted stereo depth estimation. In *CVPR*, pages 2189–2199, 2020. 5, 2

[24] Yixing Lao, Xiaogang Xu, Zhipeng Cai, Xihui Liu, and Hengshuang Zhao. CorresNeRF: Image correspondence priors for neural radiance fields. In *NeurIPS*, 2023. 2

[25] Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. In *CVPR*, 2023. 1, 2

[26] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '87, page 163–169, New York, NY, USA, 1987. Association for Computing Machinery. 2

[27] Wenjie Luo, Alexander G. Schwing, and Raquel Urtasun. Efficient deep learning for stereo matching. In *CVPR*, 2016. 2

[28] Xuan Luo, Jia-Bin Huang, Richard Szeliski, Kevin Matzen, and Johannes Kopf. Consistent video depth estimation. *ACM TOG*, 39(4):71–1, 2020. 5, 2

[29] I. Melekhov, J. Kannala, and E. Rahtu. Image patch matching using convolutional descriptors with euclidean distance. In *Proc. ACCVW*, 2016. 2

[30] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 2

[31] Anastasiya Mishchuk, Dmytro Mishkin, Filip Radenović, and Jiři Matas. Working hard to know your neighbor's margins: local descriptor learning loss. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 4829–4840. Curran Associates Inc., 2017. 2

[32] Zak Murez, Tarrence Van As, James Bartolozzi, Ayan Sinha, Vijay Badrinarayanan, and Andrew Rabinovich. Atlas: End-to-end 3d scene reconstruction from posed images. In *ECCV*, pages 414–431. Springer, 2020. 5, 2

[33] Michael Niemeyer, Jonathan T. Barron, Ben Mildenhall, Mehdi S. M. Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[34] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8024–8035. Curran Associates, Inc., 2019. 5

[35] Lukas Radl, Michael Steiner, Mathias Parger, Alexander Weinrauch, Bernhard Kerbl, and Markus Steinberger. StopThePop: Sorted Gaussian Splatting for View-Consistent Real-time Rendering. *ACM Transactions on Graphics*, 4(43), 2024. 3

[36] Marie-Julie Rakotosaona, Fabian Manhardt, Diego Martin Arroyo, Michael Niemeyer, Abhijit Kundu, and Federico Tombari. Nerfmeshing: Distilling neural radiance fields into geometrically-accurate 3d meshes. In *Proc. of the International Conf. on 3D Vision (3DV)*, 2024. 2

[37] Xuqian Ren, Wenjia Wang, Dingding Cai, Tuuli Tuominen, Juho Kannala, and Esa Rahtu. Mushroom: Multi-sensor hybrid room dataset for joint 3d reconstruction and novel view synthesis. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 4508–4517, 2024. 3, 5, 1

[38] Barbara Roessle, Jonathan T. Barron, Ben Mildenhall, Pratul P. Srinivasan, and Matthias Nießner. Dense depth priors for neural radiance fields from sparse input views. In *CVPR*, 2022. 2

[39] Darius Rückert, Linus Franke, and Marc Stamminger. Adop: Approximate differentiable one-pixel point rendering. *ACM TOG*, 41(4):1–14, 2022. 2

[40] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 2, 3, 5, 7, 1

[41] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*, 2016. 2, 3, 5, 7

[42] Ayan Sinha, Zak Murez, James Bartolozzi, Vijay Badrinarayanan, and Andrew Rabinovich. Deltas: Depth estimation by learning triangulation and densification of sparse points. In *ECCV*, pages 104–121. Springer, 2020. 5, 2

[43] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM siggraph 2006 papers*, pages 835–846. Association for Computing Machinery (ACM), 2006. 2

[44] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 3

[45] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Justin Kerr, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, David McAllister, and Angjoo Kanazawa. Nerfstudio: A modular framework for neural radiance field development. In *ACM SIGGRAPH 2023 Conference Proceedings*, SIGGRAPH '23, 2023. 2, 5, 6, 7, 8, 1, 4, 9

[46] Jiaxiang Tang, Hang Zhou, Xiaokang Chen, Tianshu Hu, Errui Ding, Jingdong Wang, and Gang Zeng. Delicate textured mesh recovery from nerf via adaptive surface refinement. *arXiv preprint arXiv:2303.02091*, 2022. 2

[47] Zachary Teed and Jia Deng. Deepv2d: Video to depth with differentiable structure from motion. *arXiv preprint arXiv:1812.04605*, 2018. 5, 2

[48] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. Ref-NeRF: Structured view-dependent appearance for neural radiance fields. In *CVPR*, 2022. 1

[49] Jingwen Wang, Tymoteusz Bleja, and Lourdes Agapito. Go-surf: Neural feature grid optimization for fast, high-fidelity rgb-d surface reconstruction. In *2022 International Conference on 3D Vision (3DV)*, pages 433–442. IEEE, 2022. 5

[50] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021. 1, 2, 3

[51] Yi Wei, Shaohui Liu, Yongming Rao, Wang Zhao, Jiwen Lu, and Jie Zhou. Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo. In *ICCV*, pages 5610–5619, 2021. 2, 5

[52] Haolin Xiong, Sairisheek Muttukuru, Rishi Upadhyay, Pradyumna Chari, and Achuta Kadambi. Sparsegs: Real-time 360° sparse view synthesis using gaussian splatting. *Arxiv*, 2023. 8

[53] 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. 1, 4

[54] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. In *NeurIPS*, 2021. 2

[55] Vickie Ye, Ruilong Li, Justin Kerr, Matias Turkulainen, Brent Yi, Zhuoyang Pan, Otto Seiskari, Jianbo Ye, Jeffrey

Hu, Matthew Tancik, and Angjoo Kanazawa. gsplat: An open-source library for gaussian splatting. *arXiv preprint arXiv:2409.06765*, 2024. 5

[56] Vickie Ye, Matias Turkulainen, and the Nerfstudio team. gsplat. 1

[57] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *ICCV*, 2023. 3, 5

[58] Zehao Yu, Anpei Chen, Bozidar Antic, Songyou Peng, Apratim Bhattacharyya, Michael Niemeyer, Siyu Tang, Torsten Sattler, and Andreas Geiger. Sdfstudio: A unified framework for surface reconstruction, 2022. 5, 6, 1

[59] Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction. *NeurIPS*, 2022. 2, 4, 5, 6, 7, 1, 8, 9

[60] Jingyang Zhang, Yao Yao, Shiwei Li, Zixin Luo, and Tian Fang. Visibility-aware multi-view stereo network. *British Machine Vision Conference (BMVC)*, 2020. 2, 8

[61] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv preprint arXiv:1801.09847*, 2018. 4

[62] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, pages 1851–1858, 2017. 5, 2

Supplementary Material

Matias Turkulainen*[1] Xuqian Ren*[2] Iaroslav Melekhov[3] Otto Seiskari[4] Esa Rahtu[2]
Juho Kannala[3,4]
[1] ETH Zurich, [2] Tampere University, [3] Aalto University, [4] Spectacular AI

In this **supplementary material**, we provide further details regarding our baseline methods and datasets in Appendix A, definitions for our evaluation metrics and losses in Appendix B, and further quantitative and qualitative results in Appendix C and Appendix D respectively.

A. Implementation details

A.1. Baselines

We compare a variety of baseline methods for novel view synthesis, depth estimation, and mesh reconstruction.

`Nerfacto`. We use the Nerfacto model from Nerfstudio [45] version 1.0.2 in our experiments. We use default settings, disable pose optimization, and predict normals using the proposed method from Ref-NeRF [48]. We use rendered normal and depth maps for Poisson surface reconstruction.

`Depth-Nerfacto`. We use the depth supervised variant of Nerfacto with a direct loss on ray termination distribution for sensor depth supervision as described in DS-NeRF [6]. Besides this, we use the same settings as for Nerfacto.

`Neusfacto`. We use default settings provided by Neusfacto from SDFStudio [58] and use the default marching cubes algorithm for meshing.

`MonoSDF`. We use the recommended settings from MonoSDF [59] and with sensor depth and monocular normal supervision. We set the sensor depth loss multiplier to 0.1 and normal loss multiplier to 0.05. Normal predictions are obtained from Omnidata [7].

`Splatfacto`. The Splatfacto model from Nerfstudio version 1.1.3 and `gsplat` [56] version 1.0.0 serves as our baseline 3DGS model. This is a faithful re-implementation of the original 3DGS work [19]. We keep all the default settings for the baseline comparison.

`SuGaR`. We use the official SuGaR [11] source-code. The original code-base, written as an extension to the original 3DGS work [19], supports only COLMAP based datasets (that is, datasets containing a COLMAP database file). We made slight modifications to the original source-code to support non-COLMAP based formats to import camera information and poses directly from a pre-made .json files. We use default settings for training as described in [11]. We use the SDF trained variant in all experiments. We extract both the coarse and refined meshes for evaluation, although the difference in geometry metrics are small between them. We found a small inconsistency in SuGaR's normal directions for outward facing indoor datasets, which we corrected in our experiments.

In addition, we have modified the original source-code to support depth rendering, which was not possible in the original author's code release. This is achieved by replacing the CUDA backend with a variant that also includes depth rendering support.

`2DGS`. We use the official 2DGS [16] source-code. Similar to our SuGaR implementation, we made slight modifications to the original source-code to support non-COLMAP based formats to import camera information and poses directly from a pre-made .json files. We use default settings for training as described in [16] and the default meshing strategy using TSDF fusion.

`2DGS + `$\mathcal{L}_{\hat{\mathcal{D}}}$ `variant`. We implement the proposed depth regularization strategy into the official 2DGS code release. Specifically, we enable supervision and gradient flow to depths within the CUDA backend rasterizer and supervise with sensor or monocular depth estimates. Our overall optimization loss becomes $\mathcal{L} = \mathcal{L}_{\text{rgb}} + \lambda_d \mathcal{L}_d$ where λ_d is set to 0.2 and $\mathcal{L}_{\text{rgb}}$ is the original loss from [16].

A.2. Datasets

MuSHRoom. We use the official train and evaluation splits from the MuSHRoom [37] dataset. We report evaluation metrics on a) images obtained from uniformly sampling every 10 frames from the training camera trajectory and b) images obtained from a different camera trajecotry. We use the globally optimized COLMAP [40] for both evaluation sequences. We use a total of 5 million points for mesh extraction for Poisson surface reconstruction.

ScanNet++. We use the "b20a261fdf" and "8b5caf3398" scenes in our experiments. We use the iPhone sequences with COLMAP registered poses. The sequences contain 358 and 705 registered images respectively. We uniformly load every 5th frame from the sequences from which we reserve every 10th frame for evaluation.

Metric	Definition
Abs Rel	$\frac{1}{N}\sum_{i=1}^{N}\frac{\left\lvert d_i^{\text{pred}}-d_i^{\text{gt}}\right\rvert}{d_i^{\text{gt}}}$
Sq Rel	$\frac{1}{N}\sum_{i=1}^{N}\frac{\left(d_i^{\text{pred}}-d_i^{\text{gt}}\right)^2}{d_i^{\text{gt}}}$
RMSE	$\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(d_i^{\text{pred}}-d_i^{\text{gt}}\right)^2}$
RMSE log	$\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\log d_i^{\text{pred}}-\log d_i^{\text{gt}}\right)^2}$
Threshold accuracy, δ	$\frac{1}{N}\sum_{i=1}^{N}\left[\max\left(\frac{d_i^{\text{pred}}}{d_i^{\text{gt}}},\frac{d_i^{\text{gt}}}{d_i^{\text{pred}}}\right)<\delta\right]$

Table 9. **Depth Evaluation Metrics.** We show definitions for our depth evaluation metrics. d_i^{pred} and d_i^{gt} are predicted and ground-truth depths for the i-th pixel. δ is the threshold factor (*e.g.*, $\delta < 1.25$, $\delta < 1.25^2$, $\delta < 1.25^3$).

B. Definitions for metrics and losses

B.1. Depth evaluation metrics

For the ScanNet++ and MuSHRoom datasets, we follow [23,28,32,42,47,51,62] and report depth evaluation metrics, defined in Table 9. We use the Absolute Relative Distance (*Abs Rel*), Squared Relative Distance (*Sq Rel*), Root Mean Squared Error *RMSE* and its logarithmic variant *RMSE log*, and the *Threshold Accuracy* ($\delta < t$) metrics. The *Abs Rel* metric provides a measure of the average magnitude of the relative error between the predicted depth values and the ground truth depth values. Unlike the *Abs Rel* metric, the *Sq Rel* considers the squared relative error between the predicted and ground truth depth values. The *RMSE* metric calculates the square root of the average of the squared differences between the predicted and the ground-truth values, giving a measure of the magnitude of the error made by the predictions. The *RMSE log* metric is similar to *RMSE* but applied in the logarithmic domain, which can be particularly useful for very large depth values. The *Threshold accuracy* measures the percentage of predicted depth values within a certain threshold factor, δ of the ground-truth depth values.

B.2. Mesh evaluation metrics

In Table 10 we provide the definitions for mesh evaluation used throughout the text for comparing predicted and ground truth meshes. We use a threshold of $5cm$ for precision, recall, and F-scores. Furthermore, we evaluate mesh quality only within the visibility of the training camera views.

B.3. Depth losses

For depth supervision, we compare the following variants of loss functions defined in Table 11

We compare the performance of these losses as supervision in Table 15.

Metric	Definition
Accuracy	$\frac{1}{\lvert P\rvert}\sum_{\mathbf{p}\in P}\left(\min_{\mathbf{p}^*\in P^*}\lVert\mathbf{p}-\mathbf{p}^*\rVert_1\right)$
Completion	$\frac{1}{\lvert P^*\rvert}\sum_{\mathbf{p}^*\in P^*}\left(\min_{\mathbf{p}\in P}\lVert\mathbf{p}-\mathbf{p}^*\rVert_1\right)$
Chamfer-L_1	$\frac{\text{Accuracy}+\text{Completion}}{2}$
Normal Completion	$\frac{1}{\lvert P^*\rvert}\sum_{\mathbf{p}^*\in P^*}\left(\mathbf{n}_{\mathbf{p}}^T\mathbf{n}_{\mathbf{p}^*}\right)$ s.t. $\mathbf{p}=\underset{p\in P}{\text{argmin}}\lVert\mathbf{p}-\mathbf{p}^*\rVert_1$
Normal-Consistency	$\frac{\text{Normal-Acc+Normal-Comp}}{2}$
Precision	$\frac{1}{\lvert P\rvert}\sum_{\mathbf{p}\in P}\left(\min_{\mathbf{p}^*\in P^*}\lVert\mathbf{p}-\mathbf{p}^*\rVert_1<5cm\right)$
Recall	$\frac{1}{\lvert P^*\rvert}\sum_{\mathbf{p}^*\in P^*}\left(\min_{\mathbf{p}\in P}\lVert\mathbf{p}-\mathbf{p}^*\rVert_1<5cm\right)$
F-score	$\frac{2\cdot\text{Precision}\cdot\text{Recall}}{\text{Precision}+\text{Recall}}$

Table 10. **Mesh Evaluation Metrics**. P and P^* are the point clouds sampled from the predicted and the ground truth mesh. n_p is the normal vector at point $\mathbf{p}$.

Loss	Definition
$\mathcal{L}_{\text{MSE}}$	$\frac{1}{\lvert\hat{D}\rvert}\sum(\hat{D}-D)^2$
$\mathcal{L}_1$	$\frac{1}{\lvert\hat{D}\rvert}\sum\lVert\hat{D}-D\rVert_1$
$\mathcal{L}_{\text{LogL1}}$	$\frac{1}{\lvert\hat{D}\rvert}\sum\log(1+\lVert\hat{D}-D\rVert_1)$
$\mathcal{L}_{\text{HuberL1}}$	$\begin{cases}\lVert D-\hat{D}\rVert_1, & \text{if } \lVert D-\hat{D}\rVert_1\le\delta,\\ \frac{(D-\hat{D})^2+\delta^2}{2\delta}, & \text{otherwise.}\end{cases}$
$\mathcal{L}_{\text{DSSIML1}}$	$\alpha\frac{1-\text{SSIM}(I,\hat{I})}{2}+(1-\alpha)\lvert I-\hat{I}\rvert$
$\mathcal{L}_{\text{EAS}}$	$g_{\text{rgb}}\frac{1}{\lvert\hat{D}\rvert}\sum\lVert\hat{D}-D\rVert_1$
$\mathcal{L}_{\hat{D}}$	$g_{\text{rgb}}\frac{1}{\lvert\hat{D}\rvert}\sum\log(1+\lVert\hat{D}-D\rVert_1)$

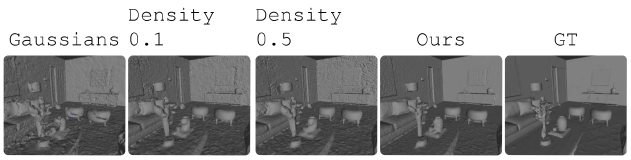
Table 11. **Depth Regularization Objectives.** We show the definitions for various depth objectives. Here, $\delta = 0.2\max(\lVert D-\hat{D}\rVert_1)$, $g_{\text{rgb}} = \exp(-\nabla I)$, $D/\hat{D}$ are the ground truth and rendered depths, and $I/\hat{I}$ is the ground truth/rendered RGB image.

C. Additional quantitative results

Here we provide additional quantitative results for DN-Splatter. We provide a comparison of Poisson meshing strategies, comparison of depth estimation quality with ground truth Faro scanner data, as well as further ablations on depth loss variants.

C.1. Mesh extraction techniques

We investigate various Poisson meshing techniques. In Table 12, we demonstrate that extracting oriented point sets from optimized depth and normal maps results in smoother and more realistic reconstructions compared to other methods. We report mesh evaluation metrics for these different techniques. We compare several approaches: directly using trained Gaussian means and normals for Poisson meshing (total of 512k Gaussians); extraction of surface density at levels 0.1 and 0.5 by projecting rays from camera views and querying scene intersections based on local density values, as proposed in SuGaR [11]; and back-projection of optimized depth and normal maps. All models were trained with our depth and normal regularization. To ensure a fair comparison, we set the total number of extracted points to 500k for both the surface density and back-projection methods.



	Acc. ↓	Comp. ↓	C-L_1 ↓	NC ↑	F-score ↑
Gaussians	.0206	.0412	.0309	.9091	.9117
SuGaR [11]: density 0.1	.0130	.0357	.0243	.9301	.9275
SuGaR [11]: density 0.5	.0083	**.0304**	**.0193**	.9309	**.9325**
Back-projection (ours)	**.0074**	.0312	.0194	**.9428**	.9310

Table 12. **Ablation of Poisson mesh extraction techniques: Replica.** We compare naive Gaussian-based meshing, the meshing strategy proposed in SuGaR [11], and our back-projection approach. All models were trained using the proposed depth and normal objectives.

C.2. Depth estimation compared to Faro scanner ground truth

In Table 13, we show the depth evaluation performance of our proposed regularization scheme on the MuSHRoom dataset, evaluated against ground truth Faro lidar scanner data instead of the low-resolution iPhone depths. This corresponds to Table 3 from the main paper, which compares depth metrics on iPhone depth captures for the same scenes and baselines. When comparing to laser scanner depths, our method still out performs other baseline methods on depth estimation.

C.3. Additional depth comparisons

We consider the performance of DN-Splatter within sparse view setting guided by only monocular depth estimates. We test on the large scale Tanks & Temples scene in Table 14. We consider training with dense and sparse captures and conclude that although monocular depth supervision in dense captures provides minimal improvements, the increase in novel view synthesis metrics under sparse view settings is notable. Lastly, in Table 15 we compare the performance of various depth losses described in Section B.3 on depth estimation and novel view synthesis. There are several interesting observations. First, the logarithmic depth loss L_{LogL1} outperforms other popular variants like L_{L1} or L_{MSE} on depth and RGB synthesis. Second, the gradient-aware logarithmic depth variant $L_{\hat{D}}$ outperforms the simpler variant, validating our assumption that captured sensor depths, like those from iPhone cameras, tend to contain noise and inaccuracies at edges or sharp boundaries. Therefore, the gradient-aware variant mitigates these inaccurate sensor readings.

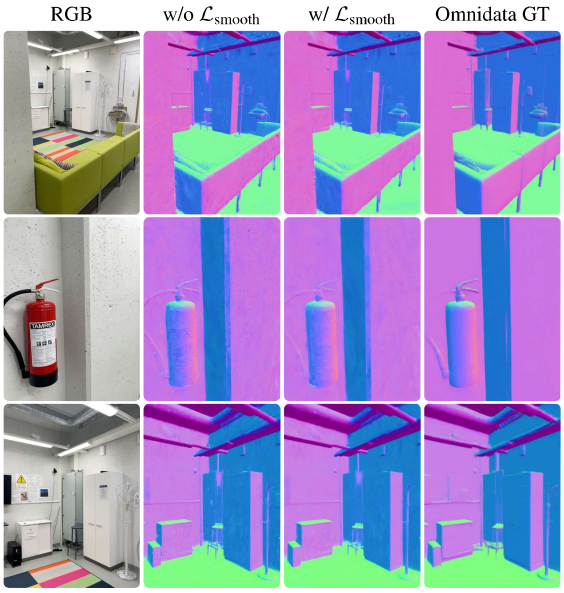


Figure 7. **Qualitative comparison of normal smoothing prior.** We visualize normal estimates with and without our $\mathcal{L}_{\text{smooth}}$ smoothing prior on the 'VR Room' scene from MuSHRoom dataset.

D. Additional qualitative results

D.1. Normal smoothing loss

We visualize the impact of $\mathcal{L}_{\text{smooth}}$ prior on rendered normal estimates in Fig. 7. We achieve smoother predictions with the prior.

D.2. 2DGS vs. DN-Splatter renders

In Fig. 8 we compare novel-view and depth estimation renders using baseline Splatfacto and 2DGS models as well as a variant of 2DGS with depth supervision enabled and our method.

D.3. Mesh and NVS renders

Lastly, we provide additional qualitative results for mesh performance in Fig. 9 as well as depth and novel view renders in Fig. 10, Fig. 11, and Fig. 12, respectively.

	Sensor Depth	(a) Test within a sequence Abs Rel ↓	Sq Rel ↓	RMSE ↓	RMSE log ↓	$\delta < 1.25$ ↑	(b) Test with a different sequence Abs Rel ↓	Sq Rel ↓	RMSE ↓	RMSE log ↓	$\delta < 12.5$ ↑
Nerfacto [45]	–	14.72	19.79	61.05	13.26	88.25	14.52	18.32	63.85	13.13	88.41
Depth-Nerfacto [45]	✓	13.90	11.71	50.21	12.98	88.46	13.49	10.76	51.63	12.62	89.23
MonoSDF [59]	✓	10.90	9.87	48.74	11.27	83.48	11.00	10.98	50.92	11.37	82.62
Splatfacto (no cues) [19]	–	8.32	5.45	38.47	10.23	89.75	8.06	5.39	38.61	10.05	90.51
Splatfacto + $\mathcal{L}_{\hat{D}}$ (Ours)	✓	3.71	3.08	30.80	4.27	95.52	3.78	3.08	31.35	4.26	95.47
Splatfacto + $\mathcal{L}_{\hat{D}}$ + $\mathcal{L}_{\hat{N}}$ (Ours)	✓	**3.64**	**3.02**	**30.33**	**4.17**	**95.60**	**3.69**	**2.97**	**30.57**	**4.15**	**95.64**

Table 13. **Depth evaluation metrics compared to ground truth Faro scanner data** for the MuSHRoom dataset. Instead of evaluating on noisy captured iPhone depth maps for evaluation, we rely on more accurate depth maps reconstructed from a Faro lidar scanner. We show that our depth regularization strategy, utilizing low-resolution iPhone depths, greatly outperforms other baselines. Results are averaged over 10 scenes.

(a) We load every 3/5/8/12 views from the whole training sequence (around 260). Results are evaluated on "Courtroom" from Tanks & Temples.

Methods	load every 3 PSNR ↑	SSIM ↑	LPIPS ↓	load every 5 PSNR ↑	SSIM ↑	LPIPS ↓	load every 8 PSNR ↑	SSIM ↑	LPIPS ↓	load every 12 PSNR ↑	SSIM ↑	LPIPS ↓
Splatfacto	20.68	.7445	.1921	18.50	.6991	.2110	16.86	.6459	.2474	14.76	.5580	.3332
Ours + Zoe-Depth [3]	20.88	.7518	.1833	19.58	.7118	.2007	**17.60**	**.6568**	**.2433**	15.90	.5835	.2971
Ours + DepthAnything [53]	**20.91**	**.7528**	**.1830**	**19.60**	**.7153**	**.1997**	17.44	**.6568**	.2456	**16.24**	**.5902**	**.2924**

(b) We load every 5/8/12/20 views from the whole training sequence (around 270). Results are evaluated on "8b5caf3398" from ScanNet++ DSLR sequence.

Methods	load every 5 PSNR ↑	SSIM ↑	LPIPS ↓	load every 8 PSNR ↑	SSIM ↑	LPIPS ↓	load every 12 PSNR ↑	SSIM ↑	LPIPS ↓	load every 20 PSNR ↑	SSIM ↑	LPIPS ↓
Splatfacto	24.68	.8810	.1169	22.81	.8568	.1559	21.08	.8357	.1816	18.90	.8059	.2375
Ours + Zoe-Depth [3]	**24.72**	.8821	**.1163**	23.04	.8591	.1521	**21.81**	**.8415**	.1755	19.10	.8059	.2332
Ours + DepthAnything [53]	24.66	**.8826**	.1194	**23.21**	**.8595**	**.1507**	21.76	.8406	**.1751**	**19.51**	**.8101**	**.2321**

Table 14. **Comparison of DN-Splatter performance with monocular depth supervision**. We ablate the Zoe-Depth [3] and DepthAnything [53] monocular estimators with sparse views on the "Courtroom" sequence of Tanks & Temples advanced dataset. Monocular depth supervision aids in novel-view synthesis under sparse settings.

(a) Test split obtained by sampling uniformly every 10 frames within the training sequence.

	Depth estimation Abs Rel ↓	Sq Rel ↓	RMSE ↓	RMSE log ↓	$\delta < 1.25$ ↑	Novel view synthesis PSNR ↑	SSIM ↑	LPIPS ↓
$\mathcal{L}_{\text{MSE}}$	.0587	.0229	.2313	.0618	.9534	22.32	.7995	.1653
$\mathcal{L}_1$	.0419	.0233	.2286	.0435	.9629	22.46	.8041	.1594
$\mathcal{L}_{\text{DSSIML1}}$	.0476	.0331	.2773	.0523	.9476	21.77	.7802	.1879
$\mathcal{L}_{\text{LogL1}}$	.0430	.0267	.2414	.0444	.9609	22.48	**.8053**	**.1580**
$\mathcal{L}_{\text{HuberL1}}$	.0536	.0239	.2335	.0561	.9579	22.39	.8017	.1625
$\mathcal{L}_{\text{EAS}}$	.0954	.0572	.3581	.1103	.8726	22.18	.7951	.1780
$\mathcal{L}_{\hat{D}}$ (Ours)	**.0338**	**.0212**	**.2170**	**.0350**	**.9691**	**22.49**	.8031	.1630

(b) Test split obtained from a different camera trajectory with no overlap with the training sequence.

	Depth estimation Abs Rel ↓	Sq Rel ↓	RMSE ↓	RMSE log ↓	$\delta < 1.25$ ↑	Novel view synthesis PSNR ↑	SSIM ↑	LPIPS ↓
$\mathcal{L}_{\text{MSE}}$	.0572	.0282	.2506	.0570	.9585	19.37	.7088	.2329
$\mathcal{L}_1$	.0449	**.0248**	.2364	.0449	**.9639**	19.45	.7164	.2253
$\mathcal{L}_{\text{DSSIML1}}$	.0482	.0330	.2775	.0527	.9495	18.98	.7040	.2430
$\mathcal{L}_{\text{LogL1}}$	.0451	.0269	.2454	.0453	.9629	19.50	.7183	**.2228**
$\mathcal{L}_{\text{HuberL1}}$	.0526	.0267	.2483	.0533	.9617	19.45	.7128	.2285
$\mathcal{L}_{\text{EAS}}$	.0724	.0442	.3142	.0819	.9329	19.30	.7108	.2351
$\mathcal{L}_{\hat{D}}$ (Ours)	**.0427**	.0252	**.2335**	**.0420**	.9632	**19.53**	**.7187**	.2286

Table 15. **Ablation on depth losses** on the MuSHRoom dataset. We consider various depth losses as defined in Appendix B.3 and their impact on depth estimation and novel view synthesis. We achieve the best performance with our proposed edge-aware $\mathcal{L}_{\hat{D}}$ loss.

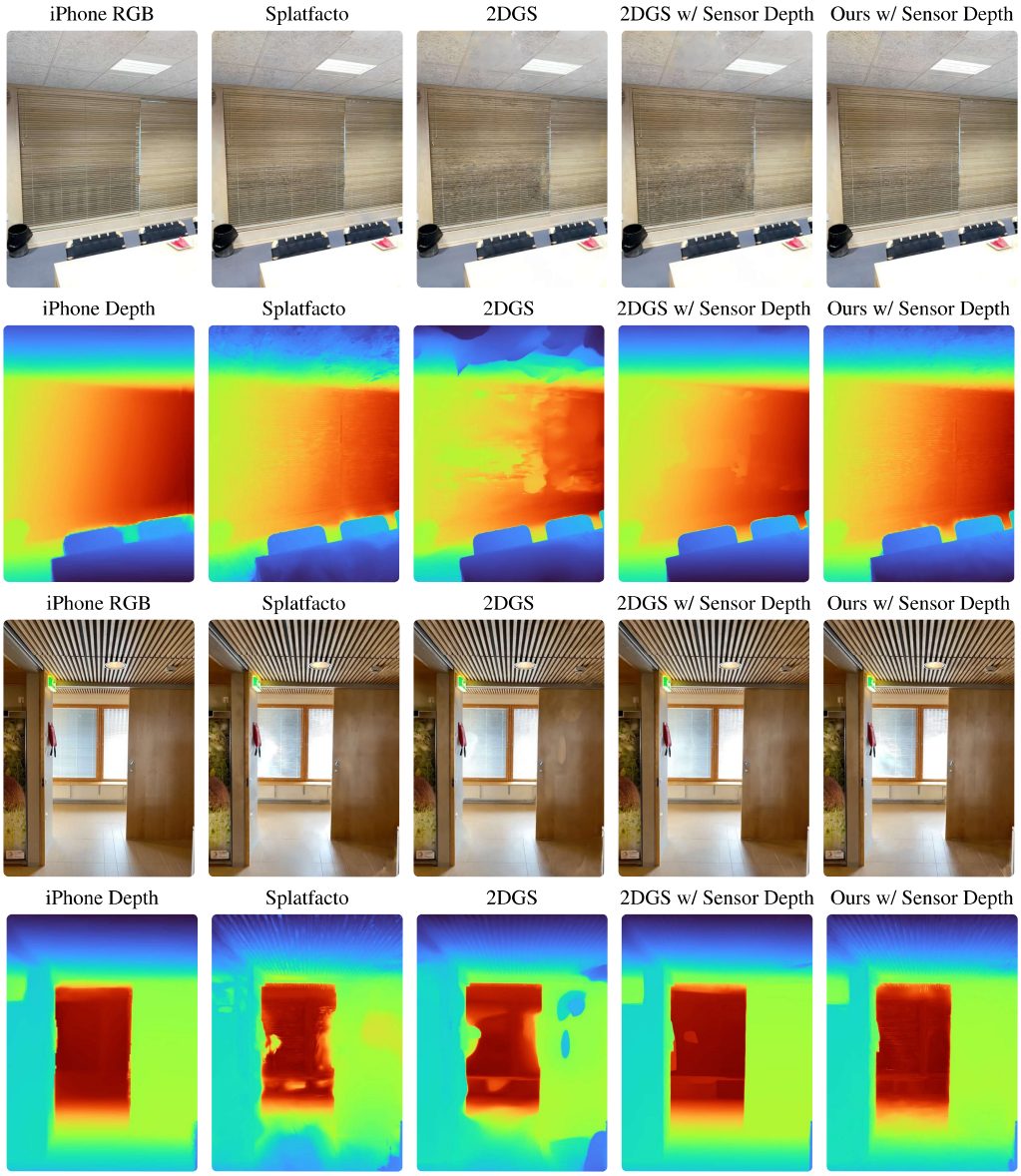
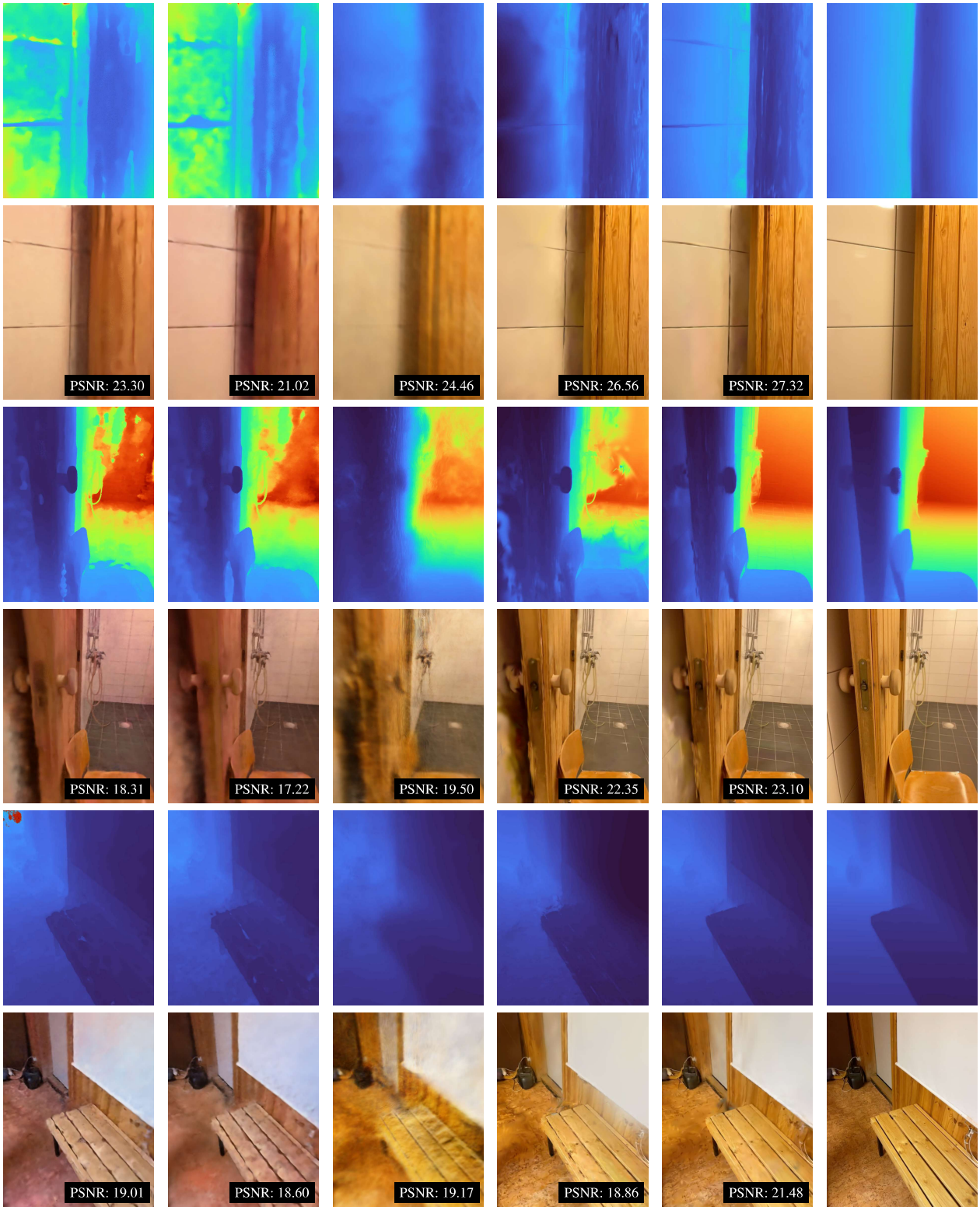


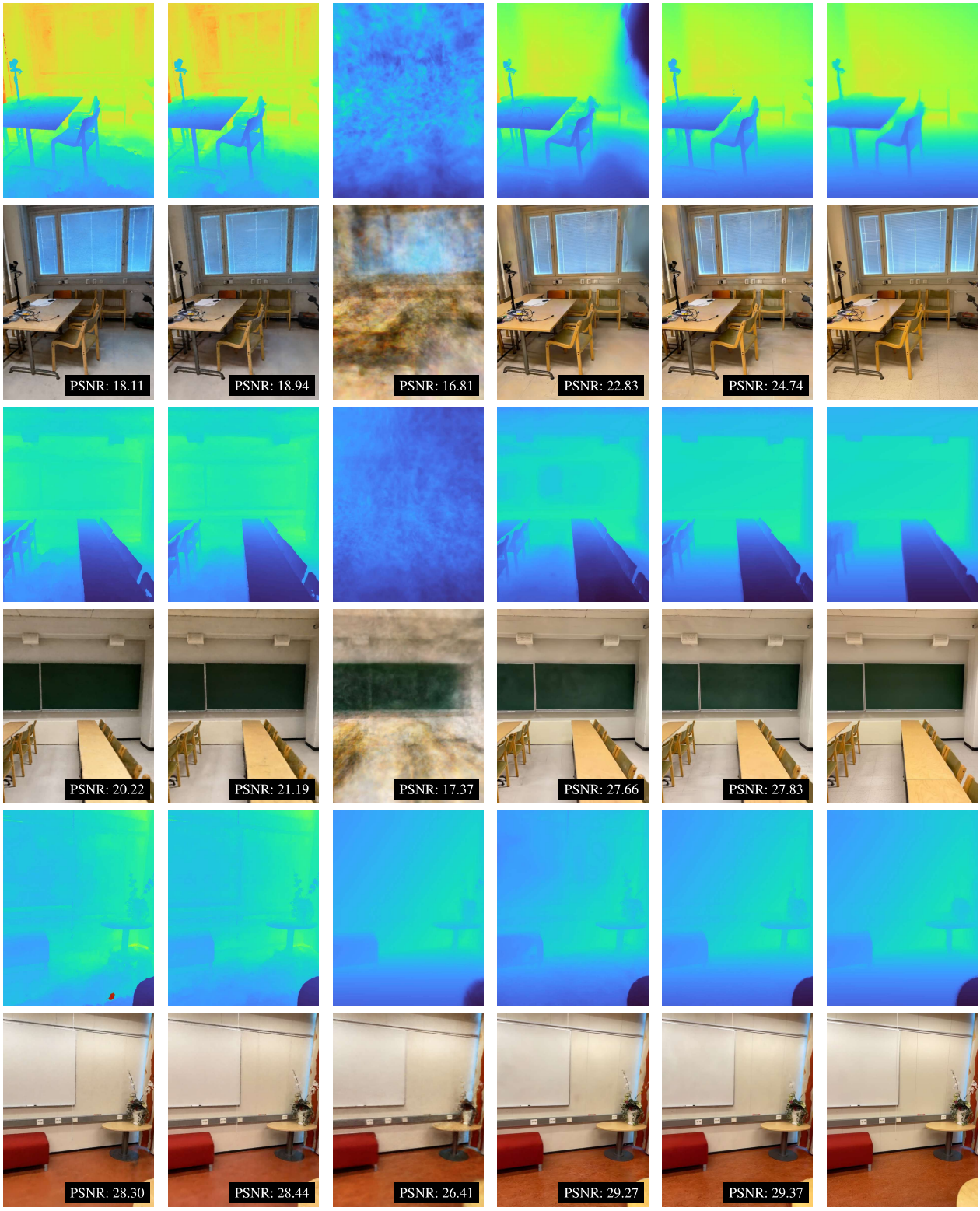
Figure 8. **Qualitative comparison between 2DGS and DN-Splatter.** We supervise both 2DGS and our method with the $\mathcal{L}_{\hat{D}}$ regularization loss and visualize novel-view and depth renders from the 'Honka' and 'Kokko' scenes from the MuSHRoom dataset. We note that DN-Splatter obtains higher metrics in both novel-view and mesh reconstruction metrics; whilst 2DGS obtains more smooth depth renders.

Figure 9. **Qualitative comparison on mesh reconstruction.** Comparison of baseline methods on sequences from the MuSHRoom dataset.



Nerfacto [45] | Depth-Nerfacto [45] | MonoSDF [59] | Splatfacto [19, 45] | Ours | iPhone GT

Figure 10. **Qualitative comparison of rendered depth and RGB images.** Comparison of baseline methods on the ”sauna” sequence from the MuSHRoom dataset.



Nerfacto [45] Depth-Nerfacto [45] MonoSDF [59] Splatfacto [19, 45] Ours iPhone GT

Figure 11. **Qualitative comparison of rendered depth and RGB images.** Comparison of baseline methods on the ”classroom” and ”coffee room” sequences from the MuSHRoom dataset.

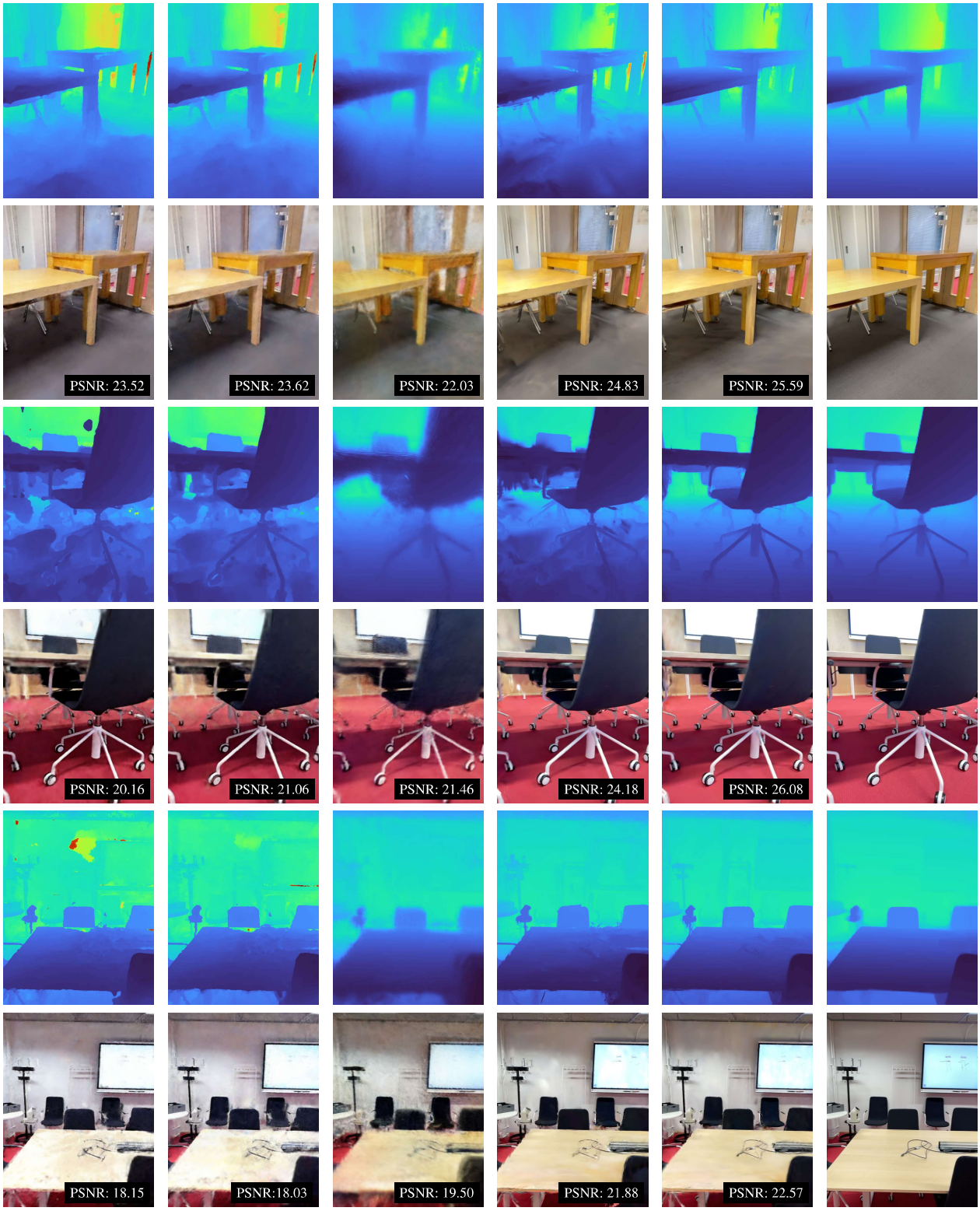


Figure 12. **Qualitative comparison of rendered depth and RGB images.** Comparison of baseline methods on the ”koivu” sequence from the MuSHRoom dataset.