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

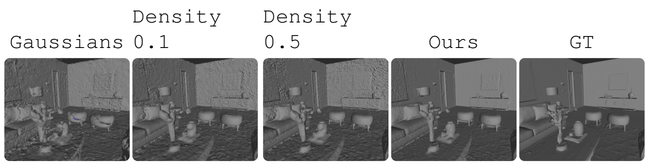
Table 11. **Depth Regularization Objectives.** We show the definitions for various depth objectives. Here, $\delta = 0.2\max(\lVert D-\hat{D}\rVert_1)$, $g_{\text{rgb}} = \exp(-\nabla I)$, $D/\hat{D}$ are the ground truth and rendered depths, and $I/\hat{I}$ is the ground truth/rendered RGB image.

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.

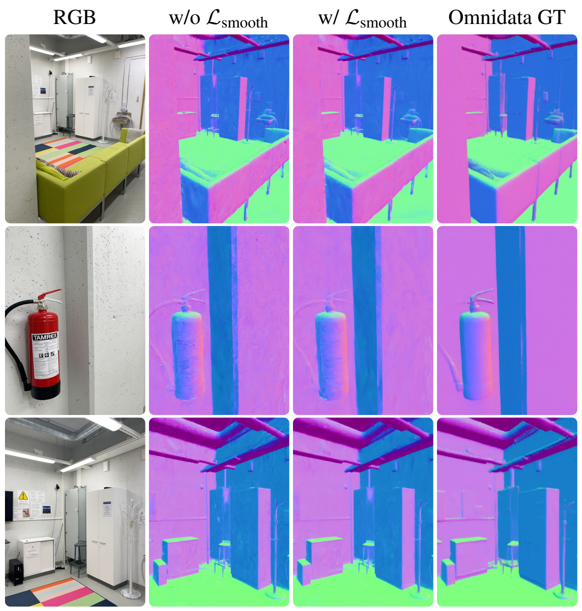

RGB
w/o $\mathcal{L}_{\text{smooth}}$
w/ $\mathcal{L}_{\text{smooth}}$
Omnidata GT

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.

		(a) Test within a sequence					(b) Test with a different sequence				
	Sensor Depth	Abs Rel ↓	Sq Rel ↓	RMSE ↓	RMSE log ↓	$\delta < 1.25$ ↑	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)	✓	<u>3.71</u>	<u>3.08</u>	<u>30.80</u>	<u>4.27</u>	<u>95.52</u>	<u>3.78</u>	<u>3.08</u>	<u>31.35</u>	<u>4.26</u>	<u>95.47</u>
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			load every 5			load every 8			load every 12		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	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			load every 8			load every 12			load every 20		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	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					Novel view synthesis		
	Abs Rel ↓	Sq Rel ↓	RMSE ↓	RMSE log ↓	$\delta < 1.25$ ↑	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					Novel view synthesis		
	Abs Rel ↓	Sq Rel ↓	RMSE ↓	RMSE log ↓	$\delta < 1.25$ ↑	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.

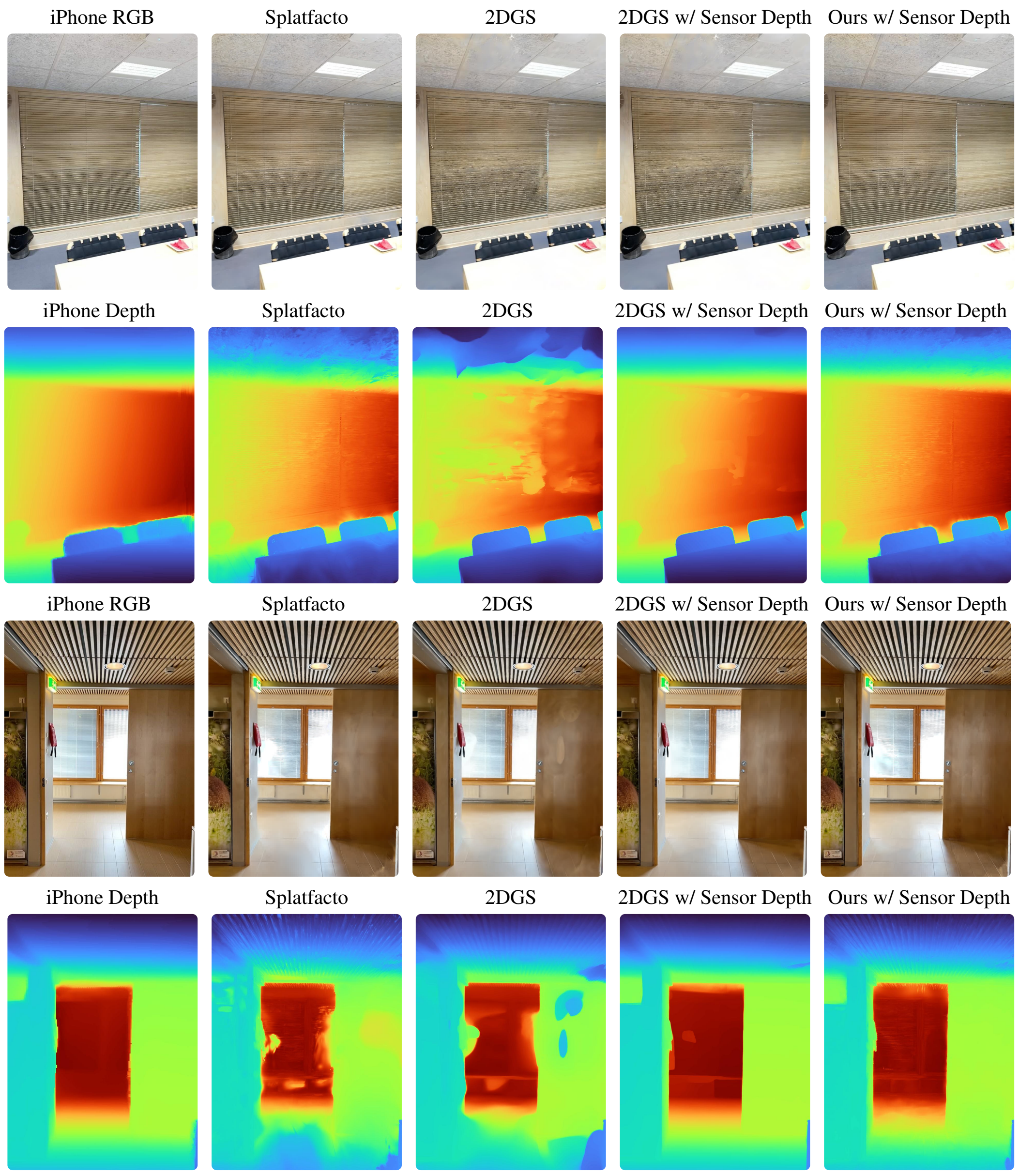


Figure 8. **Qualitative comparison between 2DGS and DN-Splatter.** We supervise both 2DGS and our method with the $\mathcal{L}_{\hat{D}}$ regularization loss and visualize novel-view and depth renders from the 'Honka' and 'Kokko' scenes from the MuSHRoom dataset. We note that DN-Splatter obtains higher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

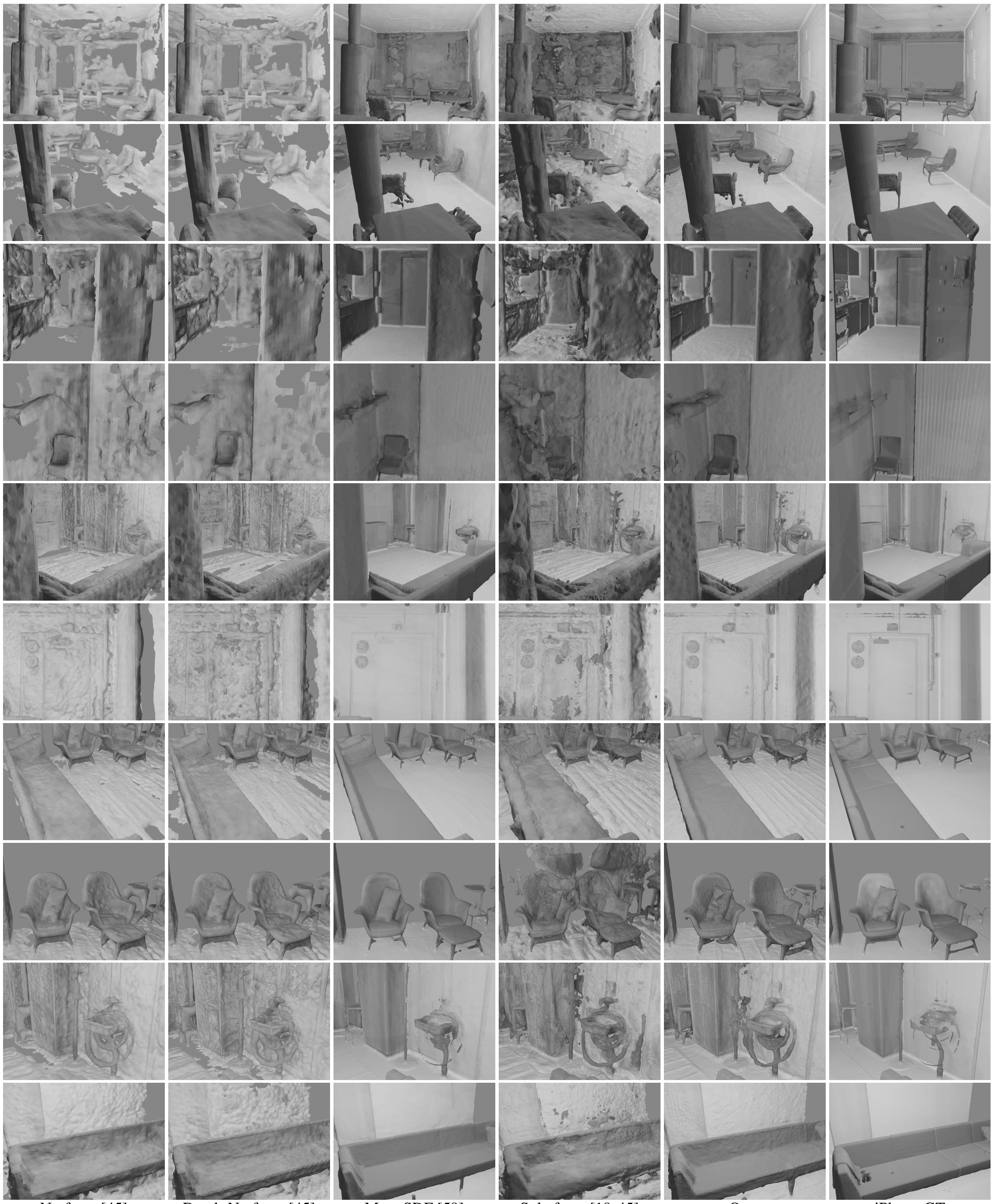


Figure 9. **Qualitative comparison on mesh reconstruction.** Comparison of baseline methods on sequences from the MuSHRoom dataset.

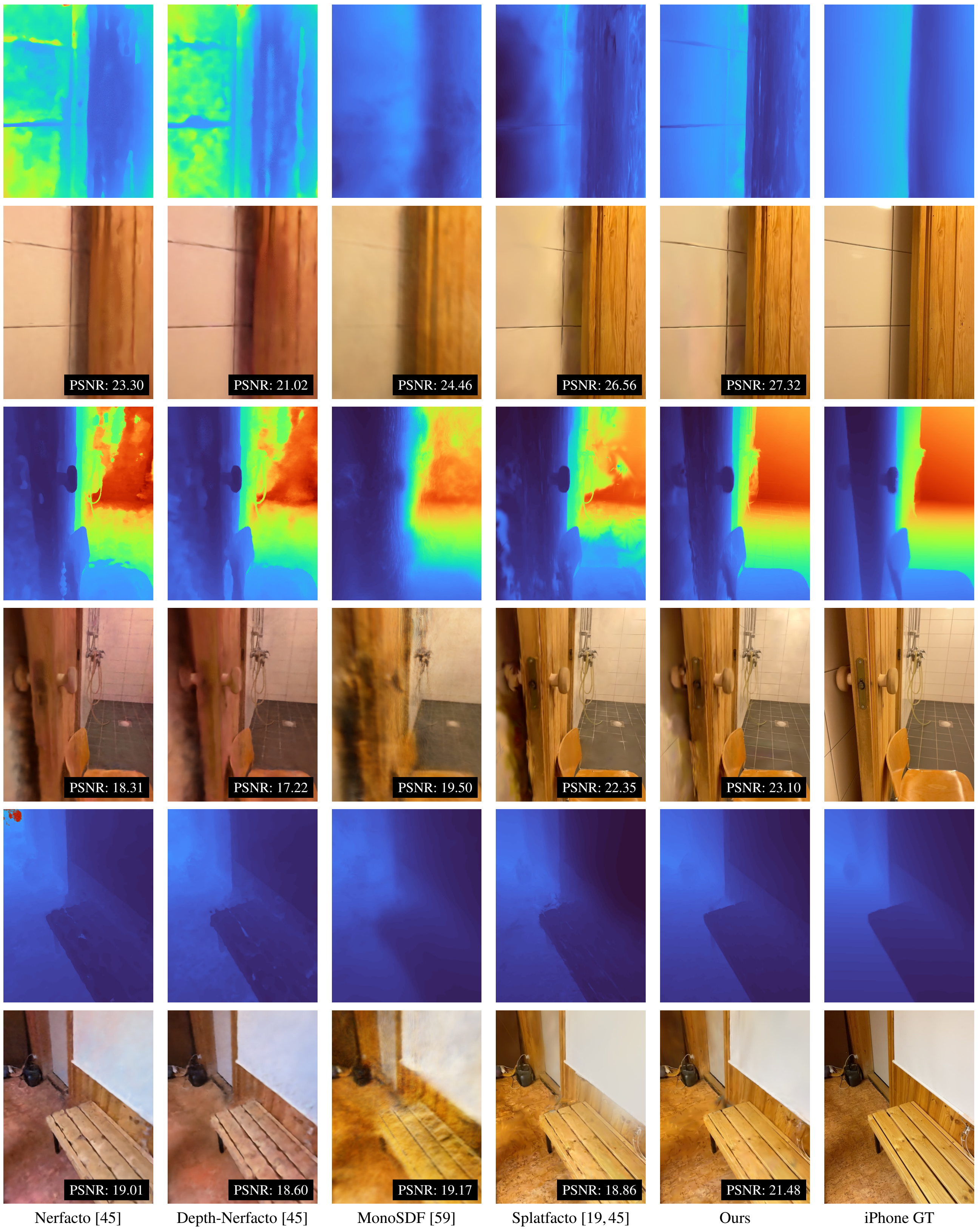


Figure 10. **Qualitative comparison of rendered depth and RGB images.** Comparison of baseline methods on the ”sauna” sequence from the MuSHRoom dataset.

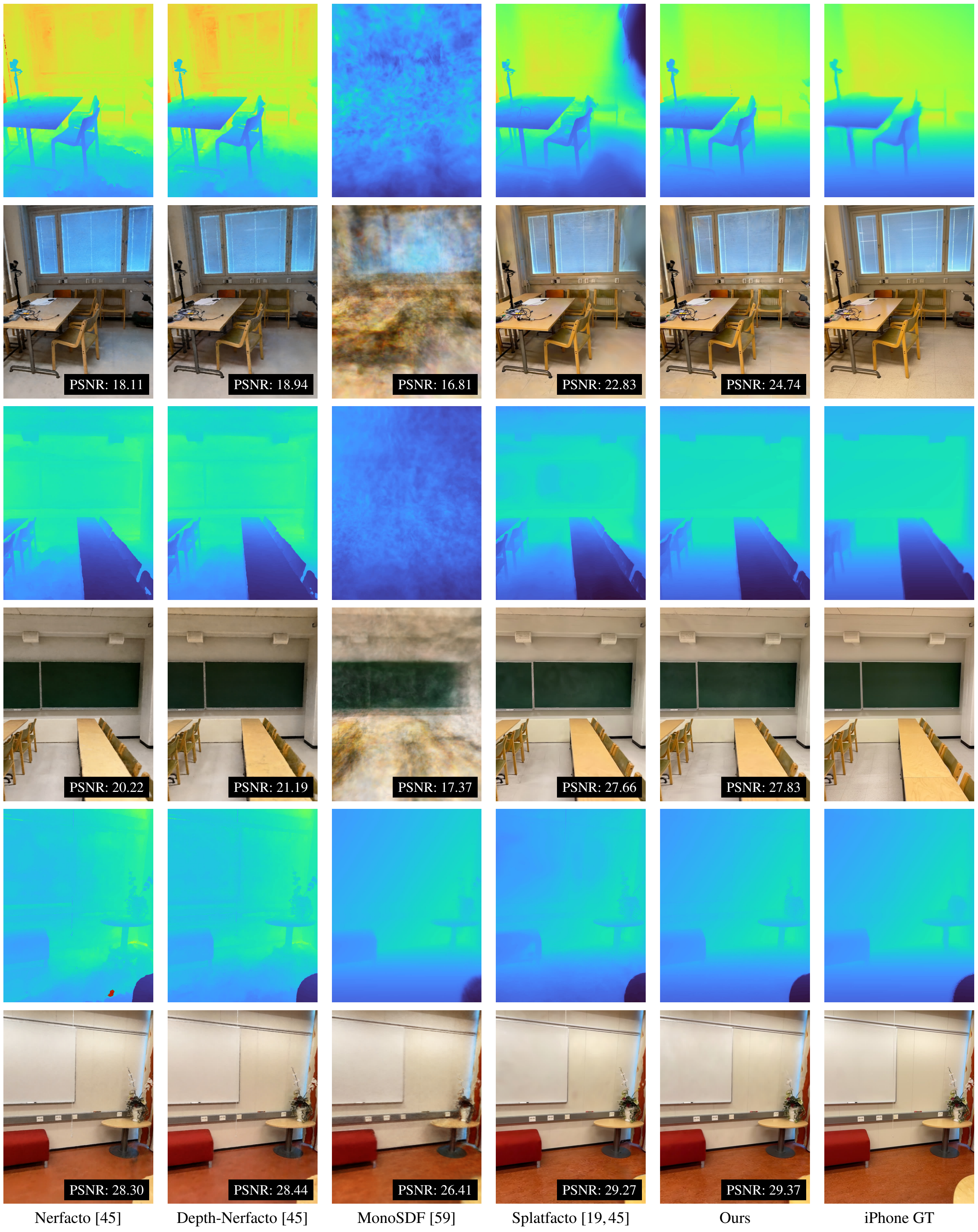


Figure 11. **Qualitative comparison of rendered depth and RGB images.** Comparison of baseline methods on the ”classroom” and ”coffee room” sequences from the MuSHRoom dataset.

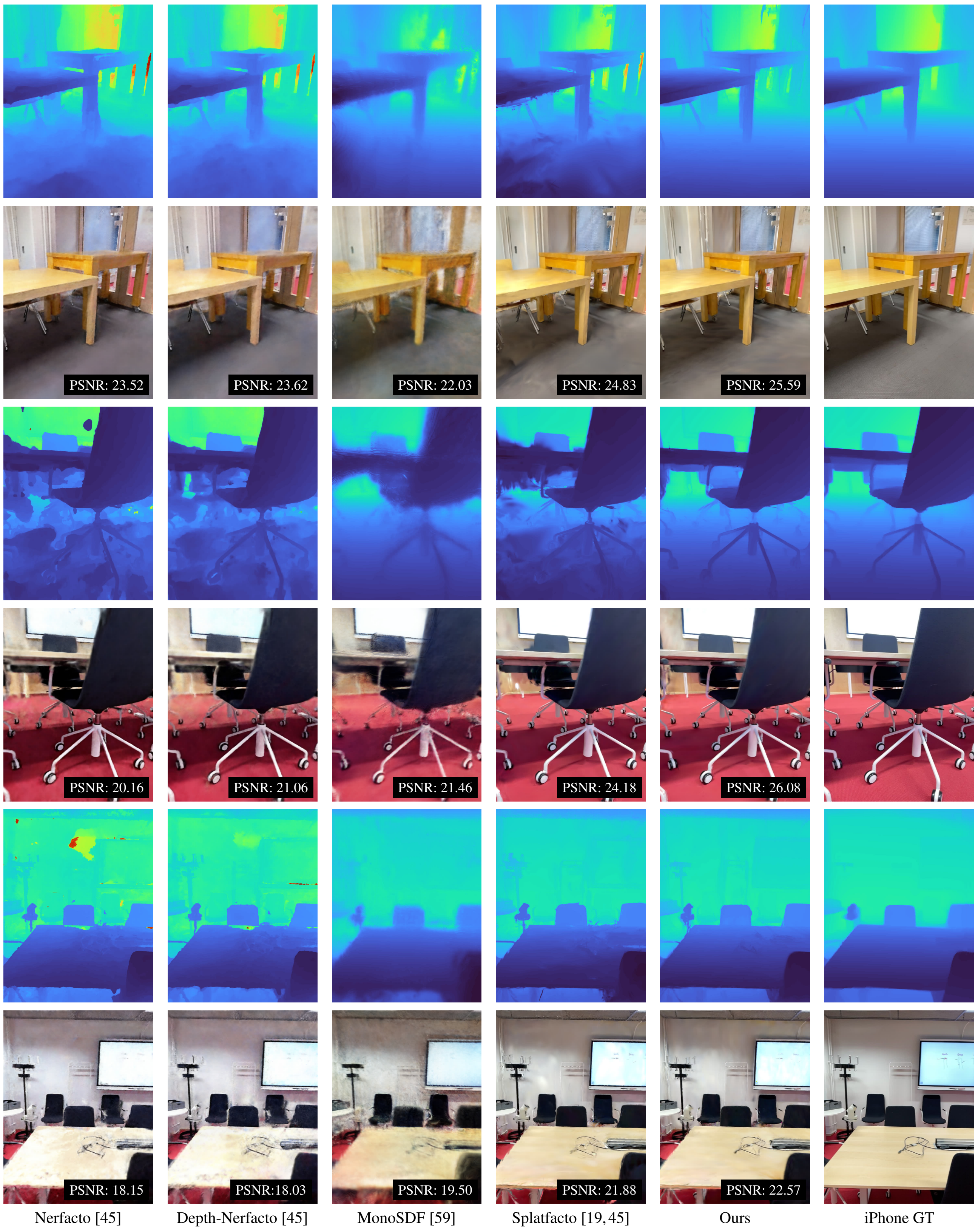


Figure 12. **Qualitative comparison of rendered depth and RGB images.** Comparison of baseline methods on the ”koivu” sequence from the MuSHRoom dataset.