

# SceneRF: Self-Supervised Monocular 3D Scene Reconstruction with Radiance Fields — Supplementary Material —

Anh-Quan Cao, Raoul de Charette  
Inria

<https://astra-vision.github.io/SceneRF>

In this supplementary material, we begin by discussing the broader impact and ethics of our work. We then present additional ablations in Sec. 1 along with further implementation details in Sec. 3. After that, we demonstrate in Sec. 2 that  $\mathcal{L}_{\text{reproj}}$  improves all baselines. Details of the baselines are presented in Sec. 4. Finally, we show additional qualitative results in Sec. 5. **The supplementary video**, which is available at <https://astra-vision.github.io/SceneRF/>, allows better evaluation of our method.

**Broader impact, Ethics.** The promotion of self-supervised monocular 3D reconstruction contributes to alleviating the needs of costly data acquisition and labeling campaigns. On the long term, this also paves the way to 3D algorithms training directly on video sequences – easier to collect and significantly more diverse than existing 3D datasets. A by-product is that it would contribute to improving generalization of 3D reconstruction. While there are no ethical concerns specific to our proposed method, we note that all methods estimating 3D from 2D are far less precise than those leveraging depth sensors (e.g., lidar, depth cameras, stereo, etc.). When it comes to safety-critical applications, like autonomous driving, we argue for use of redundant sensors.

## 1. Additional ablations

**Sampling strategy.** In Tab. 1 we illustrate the effect of varying step ( $\rho$ ) and angle ( $\phi$ ) when sampling novel depths/views in our scene reconstruction scheme (main paper, Sec. 3.4). As noted in Sec. 4 of the main paper, the views are sampled up to a distance of 10m on SemanticKITTI [2] and 2m on BundleFusion [7]. Increasing the number of synthesized depth maps (i.e., reducing the step  $\rho$ ) and varying the angle ( $\phi$ ) generally improves the IoU score. However, excessively large angle ( $\geq 20^\circ$  on SemanticKITTI [2],  $\geq 30^\circ$  on BundleFusion [2]) tends to degrade performance since the synthesized angle diverges significantly from the available angles during training. This

step (m)	rot. (deg.)	IoU	Prec.	Rec.	step (m)	rot. (deg.)	IoU	Prec.	Rec.
<i>w/o sampling</i>					<i>w/o sampling</i>				
0.25	-10 / 0 / +10	13.73	16.98	41.78	0.1	-20 / 0 / +20	20.34	25.70	49.28
0.5	-10 / 0 / +10	<b>13.84</b>	17.28	40.96	0.2	0	18.76	<b>26.43</b>	39.28
1.0	0	13.08	18.56	30.68	0.2	-10 / 0 / +10	19.94	25.80	46.76
1.0	-10 / 0 / +10	13.40	17.27	37.43	0.2	-20 / 0 / +20	20.16	25.82	47.92
1.0	-20 / 0 / +20	13.37	16.73	39.97	0.2	-30 / 0 / +30	19.98	25.90	46.64
1.0	-30 / 0 / +30	13.24	16.40	40.73	0.3	-20 / 0 / +20	19.84	25.84	46.09
2.0	-10 / 0 / +10	13.35	17.41	36.35					

(a) SemanticKITTI [2]

(b) BundleFusion [7]

Table 1: **Sampling for reconstruction.** Performance of our reconstruction scheme when varying our sampling steps ( $\rho$ ) and angles ( $\phi$ ) (val. set). The highlighted row is our main setup.

Method	Novel depth synthesis							Novel view synthesis		
	Abs Rel↓	Sq Rel↓	RMSE↓	RMSE log↓	$\delta 1 \uparrow$	$\delta 2 \uparrow$	$\delta 3 \uparrow$	LPIPS↓	SSIM↑	PSNR↑
PixelNeRF	0.6332	2.370	1.786	0.5722	48.01	73.82	84.09	0.421	0.770	19.36
MINE	0.1712	0.085	0.368	0.2214	70.74	93.42	98.46	0.430	0.714	20.21
VisionNerf	0.6384	2.813	1.934	0.5721	59.25	78.62	84.37	<b>0.391</b>	<b>0.790</b>	19.67
SceneRF	<b>0.1581</b>	<b>0.069</b>	<b>0.330</b>	<b>0.1921</b>	<b>75.81</b>	<b>96.80</b>	<b>99.66</b>	0.404	<b>0.801</b>	<b>24.02</b>

Table 2: **BundleFusion.** Train on 6 scenes and evaluate on 2 scenes.

is particularly pertinent in the context of autonomous driving setups with front-facing cameras, where there is limited peripheral supervision.

**BundleFusion evaluation on more sequences.** For completeness, we retrain on 6 sequences (apt0, apt1, apt2, office1, office2, office3) with 2 validation sequences of unseen rooms (copyroom, office0) to show generalization. The results are shown in Tab. 2 and show that SceneRF still surpasses all baselines on all 10 metrics except LPIPS. This is on par with Tab. 1 of the main paper.

## 2. Effect of $\mathcal{L}_{\text{reproj}}$ on NeRF baselines

In Tab. 3, we apply the reprojection loss  $\mathcal{L}_{\text{reproj}}$  (Sec. 3.1.1 main paper) to all NeRF baselines, showing that

Method	$\mathcal{L}_{\text{reproj}}$	Novel depth synthesis							Novel view synthesis		
		Abs Rel↓	Sq Rel↓	RMSE↓	RMSE log↓	$\delta 1\uparrow$	$\delta 2\uparrow$	$\delta 3\uparrow$	LPIPS↓	SSIM↑	PSNR↑
PixelNeRF [16]	✗	0.2364	2.080	6.449	0.3354	65.81	85.43	92.90	0.489	0.466	15.80
	✓	<b>0.1986</b>	<b>1.544</b>	<b>5.963</b>	<b>0.3093</b>	<b>70.30</b>	<b>87.19</b>	<b>93.82</b>	<b>0.488</b>	<b>0.481</b>	<b>16.11</b>
MINE [8]	✗	0.2248	1.787	6.343	0.3283	65.87	85.52	93.30	0.448	0.496	<b>16.03</b>
	✓	<b>0.2003</b>	<b>1.599</b>	<b>6.023</b>	<b>0.3070</b>	<b>70.22</b>	<b>86.98</b>	<b>93.89</b>	<b>0.445</b>	<b>0.497</b>	15.96
VisionNerf [10]	✗	0.2054	1.490	5.841	0.3073	69.11	88.28	94.37	0.468	0.483	<b>16.49</b>
	✓	<b>0.1749</b>	<b>1.380</b>	<b>5.643</b>	<b>0.2841</b>	<b>75.77</b>	<b>89.25</b>	<b>94.58</b>	<b>0.432</b>	<b>0.488</b>	16.39

Table 3: **Reprojection loss  $\mathcal{L}_{\text{reproj}}$  on other baselines.** We apply our reprojection loss to other NeRF baselines, showing it boosts performance for all.

it improves consistently all of them. Again, we argue this is because  $\mathcal{L}_{\text{reproj}}$  enforces better density ( $\sigma$ ) in the volume rendering – which has a complementary effect with  $\mathcal{L}_{\text{rgb}}$ .

### 3. Additional implementation details

#### 3.1. Probabilistic ray sampling (PrSamp) details

For clarity, in Algorithm 1, we detail the pseudocode of the Probabilistic Ray Sampling (Sec. 3.2, main paper).

#### 3.2. 3D reconstruction details

**Fusing TSDFs.** From Sec. 3.4 of the main paper, we fuse individual TSDFs by taking the minimum of their absolute values (‘min’) instead of the more standard average of all TSDFs (‘avg’). We justify this choice, in Tab. 4 showing that using ‘min’ leads to +2.77 IoU. We argue that some surfaces may be better estimated from specific camera locations. Averaging all (‘avg’) has a smoothing effect on  $V(\cdot)$  which subsequently reduces accuracy.

**Occupancy grid.** To convert the scene TSDF volume  $V(\cdot)$  (cf. Sec. 3.4, main paper) into an occupancy grid, we first study the depth estimation error. Comparing 100 frames in Fig. 1 with sparse Lidar ground truth, we note a linear relation between error estimation and ground truth depth. This motivated us to model the occupancy grid  $O(\cdot)$  as an adaptive depth threshold:

$$O(v) = 1 \iff V(v) < \min(0.25d_v, 4.0), \quad (1)$$

with  $v$  a voxel in  $V$ , and  $d_v$  its distance to the camera origin. We arbitrarily cap the threshold to 4 meters to avoid considering all far voxels as occupied.

#### 3.3. Network architecture

For 2D features extraction, the encoder is similar to [5], which is based on a pre-trained EfficientNetB7 [15]. The spherical decoder has 5 layers, each of which doubles the input resolution and halves the feature dimension. To make up for the large amount of empty space that comes with increasing the field of view, we augment the receptive field by putting three ResNet blocks with dilation sizes of 1, 2, and

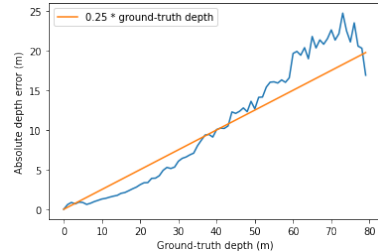


Figure 1: **Absolute depth error w.r.t. ground-truth depth.** We compute error from 100 randomly selected scenes in the training set, and observe a linear relation between error and distance.

3 in each layer. The skip connections (described in Sec. 3.3 main paper) are used between the encoder and decoder at the corresponding scale.

#### 3.4. BundleFusion – 3D ground-truth

We seek to generate for each input frame the 3D ground-truth occupancy grid. To do this, we use the camera parameters and define a volume of (4.8m, 4.8m, 3.84m) in front of the camera. The origin is set at (-2.4m, -2.4m, 0m) such that the camera is in the middle of one side of the volume and pointing inward. We then combine the depth maps to create the TSDF volume with a voxel size of 0.04m, resulting in a TSDF grid of size (120, 120, 96). Finally, the occupancy grid is obtained by thresholding the TSDF grid.

### 4. Baselines details

We re-train all baseline networks, including the novel depth/view synthesis (Sec. 4.1) and scene reconstruction baselines (Sec. 4.2). We provide the reader with additional details about our baselines.

#### 4.1. Novel depth/views baselines

We train our method and baselines using AdamW [12] optimizer on 4 Tesla V100 32g with learning rate of 1e-5 for 50 epochs. For each baseline, we rely on the recommended learning rate scheduler and number of positional encoding

---

**Algorithm 1: Probabilistic Ray Sampling.**

---

**Input :** Ray  $\mathbf{r}$ .  
**Param:** Number of Gaussians  $k$ , and  $m$  number of points per Gaussian.  
Near and far bounds:  $t_n = 0.2m$  and  $t_f = 100m$ .  
Learning rate  $lr$  of gradient descend (GD).  
**Result:** Points sampled  $\mathcal{P}$

```
1  $\mathbf{d} \leftarrow \text{dir}(\mathbf{r})$   
  // Uniform sampling (•)  
2  $\mathcal{I} \leftarrow \{\text{uniform-samp}(\text{num}=k, \text{start}=t_n, \text{end}=t_f) \times \mathbf{d}\}$  ▷ Points sampling between near and far bounds  
  // (1) Predicts Gaussians ( $\mathcal{G}$ ) with MLP  $g(\cdot)$   
3  $\mathcal{G} \leftarrow g(\{\{\mathbf{x}, \mathbf{W}(\psi(\mathbf{x}))\} \mid \forall \mathbf{x} \in \mathcal{I}\})$   
  // (2) Sample  $m$  points from Gaussians (■)  
4  $\mathcal{P} \leftarrow \emptyset$   
5 for  $i \leftarrow 1$  to  $k$  do  
6   |  $\mathcal{P} \leftarrow \mathcal{P} \cup \text{gauss-sampling}(\mathcal{G}_i, m)$   
7 end  
  // Sample 32 points uniformly (▲)  
8  $\mathcal{P} \leftarrow \mathcal{P} \cup \{\text{uniform-samp}(\text{num}=32, \text{start}=t_n, \text{end}=t_f) \times \mathbf{d}\}$   
  // (3)-(4) NeRF inference to compute densities  
9  $\sigma \leftarrow \{f(\gamma(\mathbf{x}), \mathbf{d}; \mathbf{W}(\psi(\mathbf{x})))_\sigma \mid \forall \mathbf{x} \in \mathcal{P}\}$  ▷ Densities from  $f(\cdot)$  inferences Eq.(1), main paper  
  // (5) PrSOM point-Gaussian assignment  
10  $\alpha \leftarrow \{\text{alpha-value}(s, \dots) \mid \forall s \in \sigma\}$  ▷ Compute alpha values from [13] p3  
11  $\mathcal{X} \leftarrow \{\text{PrSOM}(\mathcal{G}, \mathcal{P}, \alpha)\}$  ▷ Applies PrSOM [1]  
  // (6) Compute new Gaussians from assigned points and update  $g(\cdot)$   
12  $\mathcal{G}' \leftarrow \{(\mu(\mathcal{X}_i), \text{std}(\mathcal{X}_i)) \mid \forall i \in \mathbb{N}, 1 \leq i \leq k\}$   
13  $\mathcal{L}_{\text{gauss}} \leftarrow \frac{1}{k} \sum_i \text{Kullback-Leibler}(\mathcal{G}_i \parallel \mathcal{G}'_i)$   
14  $\mathcal{L}_{\text{surface}} \leftarrow \min_i (|\mu(\mathcal{G}'_i) - \hat{D}(\mathbf{r})|_1)$   
15  $\mathcal{L}_{\text{total}} \leftarrow \mathcal{L}_{\text{gauss}} + \mathcal{L}_{\text{surface}}$   
16  $g \leftarrow \text{GD}^{lr}(g, \nabla \mathcal{L}_{\text{total}})$  ▷ Applies gradient-descent to update  $g(\cdot)$ 
```

---

Method	IoU	Prec.	Rec.
SceneRF (avg)	11.07	11.81	63.98
SceneRF (min)	13.84	17.28	40.96

Table 4: **TSDF fusion strategy comparison on SemanticKITTI [2]**. We show that our way of extracting the TSDF described in section 3.4 is better than the traditional way of using the weighted average of TSDFs.

frequencies. For our network, since we build on PixelNeRF [16], we use its scheduler and number of frequencies. The ray batch size was 1200 for Semantic KITTI [2] and 2048 for BundleFusion [7]. The training time was around 5 days per network. Additional information about the baselines implementations is provided below.

**PixelNeRF [16]**. We use the official implementation<sup>1</sup>. Following the official sampling strategy, we sample 96 points per ray, consisting of 64 coarse points, which are used to sample 16 fine points hierarchically and 16 points around the estimated depth.

**MINE [8]**. We use the official implementation<sup>2</sup>. To balance memory cost, we use the 32 planes version.

<sup>1</sup><https://github.com/sxyu/pixel-nerf>

<sup>2</sup><https://github.com/vincentfung13/MINE>

**VisionNeRF [10]**. We use the official implementation<sup>3</sup>. To balance memory cost again, we sample 96 points (32 coarse, 64 fine) which is more than for SceneRF.

## 4.2. Scene reconstruction baselines

Only MonoScene<sup>4</sup> is a monocular baseline. To better compare with the literature, we follow the recommendation of MonoScene authors [5] and compare against the  $\text{rgb}^b$  versions of popular semantic scene completion baselines: LMSCNet<sup>5</sup> [14], 3DSketch<sup>6</sup> [6] and AICNet<sup>7</sup> [9]. More in depth, to convert the sequence of depths into 3D label to train the scene reconstruction baselines, we use the Adabin [3] model to predict the depth for each image and fuse all depths into a single TSDF volume, then turned into an occupancy grid with the same reconstruction scheme as for SceneRF (see Sec. 3.2). For all, the mesh is obtained with the traditional marching cubes [11].

## 5. Additional qualitative results

**Voxelized reconstructions on SemanticKITTI [2]**. Fig. 2 shows the voxelized reconstructions comparing our

<sup>3</sup><https://github.com/ken2576/vision-nerf>

<sup>4</sup><https://github.com/cv-rits/MonoScene>

<sup>5</sup><https://github.com/cv-rits/LMSCNet>

<sup>6</sup><https://github.com/charlesCXK/TorchSSC>

<sup>7</sup><https://github.com/waterljw/SSC>

self-supervised SceneRF with the *Depth-supervised* version of MonoScene [5] which relies on AdaBins [3] trained with Lidar ground truth. Notably, despite less supervision the predictions of SceneRF align closely with those of MonoScene in terms of overall scene architecture, object shapes, and positioning. Intriguingly, SceneRF infers the sky more accurately, attributed to the 3D consistency gained from optimizing the radiance volume. On the other hand, MonoScene struggles to predict sky arguably because AdaBins trains with lidar depth which cannot capture the sky. Nevertheless, occlusions artefacts are visible in both due to monocular supervision.

**SemanticKITTI [2].** We show additional qualitative results in Fig. 3 and Fig. 4. Overall, our method predicts smoother and finer depth maps, especially at far, which leads to a better-structured 3D scene with fewer artifacts than the baselines. When synthesizing RGB images, our approach achieves comparable results to other baseline methods.

**BundleFusion [7].** We present further qualitative results in Fig. 5, which demonstrates that SceneRF produces more accurate depth maps than other methods, especially for views that significantly differ from the input view (as observed in columns “+0.2m, -20°” and “+0.4m, +20°”). Moreover, SceneRF is the only method capable of inferring scenery that is not visible in the input field-of-view, which is particularly evident in the first and third rows.

**Generalization results on nuScenes [4].** Using the SceneRF model trained on Semantic KITTI [2], we perform inference on *unseen* nuScenes images. Additional qualitative results obtained are presented in Fig. 6. Despite the vastly different setup between the two datasets, such as differences in camera setups and locations (Germany versus USA), SceneRF is able to predict a reasonable scene structure that included *e.g.*, road, building, and vehicles.

## References

[1] Fatiha Anouar, Fouad Badran, and Sylvie Thiria. Probabilistic self-organizing map and radial basis function networks. *Neurocomputing*, 1998. 3

[2] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *ICCV*, 2019. 1, 3, 4, 6, 7, 9

[3] Shariq Farooq Bhat, Ibraheem Alhashim, and Peter Wonka. AdaBins: Depth estimation using adaptive bins. In *CVPR*, 2021. 3, 4

[4] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multi-

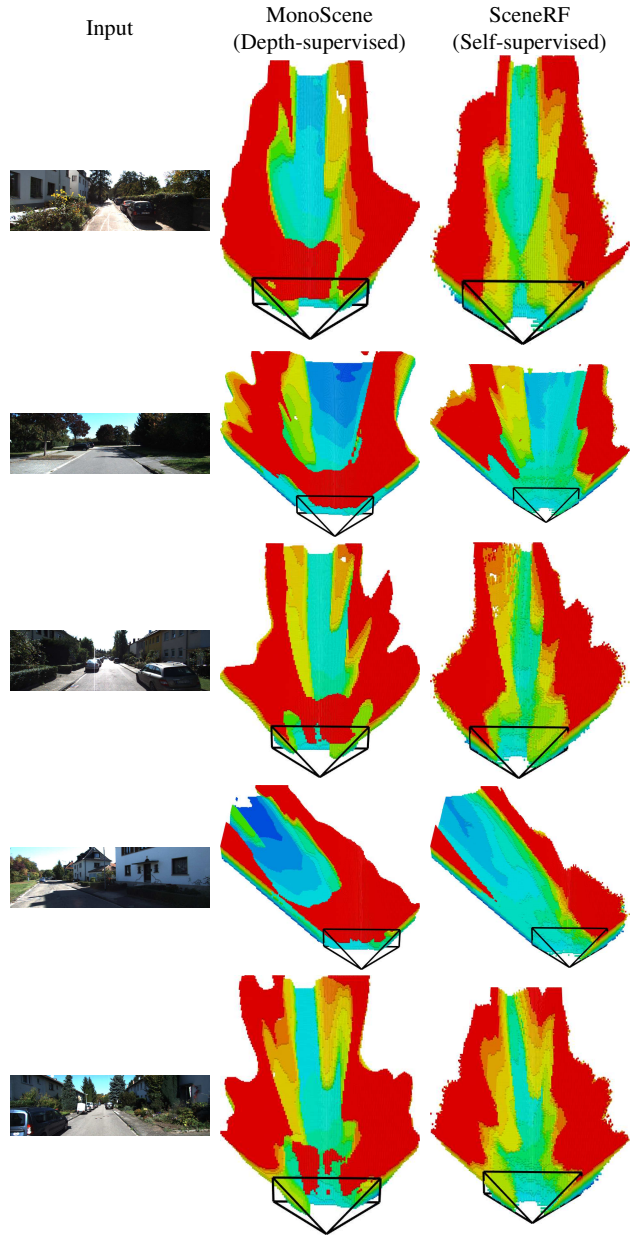


Figure 2: **Voxelized reconstructions on SemanticKITTI (val set).**

modal dataset for autonomous driving. In *CVPR*, 2020. 4, 9

[5] Anh-Quan Cao and Raoul de Charette. Monoscene: Monocular 3d semantic scene completion. In *CVPR*, 2022. 2, 3, 4

[6] Xiaokang Chen, Kwan-Yee Lin, Chen Qian, Gang Zeng, and Hongsheng Li. 3d sketch-aware semantic scene completion via semi-supervised structure prior. In *CVPR*, 2020. 3

[7] Angela Dai, Matthias Nießner, Michael Zollöfer, Shih-Wei Wang, Thibault Saito, and Christian Theobalt. Bundlesfusion: Real-time globally consistent 3d reconstruction using on-the-fly surface

- re-integration. *ACM Transactions on Graphics 2017 (TOG)*, 2017. [1](#), [3](#), [4](#), [8](#)
- [8] Jiaxin Li, Zijian Feng, Qi She, Henghui Ding, Changhu Wang, and Gim Hee Lee. Mine: Towards continuous depth mpi with nerf for novel view synthesis. In *ICCV*, 2021. [2](#), [3](#)
  - [9] Jie Li, Kai Han, Peng Wang, Yu Liu, and Xia Yuan. Anisotropic convolutional networks for 3d semantic scene completion. In *CVPR*, 2020. [3](#)
  - [10] Kai-En Lin, Yen-Chen Lin, Wei-Sheng Lai, Tsung-Yi Lin, Yichang Shih, and Ravi Ramamoorthi. Vision transformer for nerf-based view synthesis from a single input image. In *WACV*, 2023. [2](#), [3](#)
  - [11] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *SIG-GRAPH*, 1987. [3](#)
  - [12] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. [2](#)
  - [13] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *TOG*, 2019. [3](#)
  - [14] Luis Roldão, Raoul de Charette, and Anne Verroust-Blondet. Lmscnet: Lightweight multiscale 3d semantic completion. In *3DV*, 2020. [3](#)
  - [15] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019. [2](#)
  - [16] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *CVPR*, 2021. [2](#), [3](#)


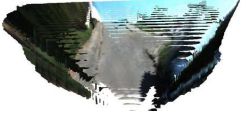
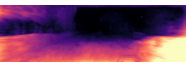
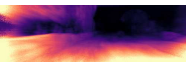
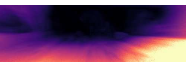


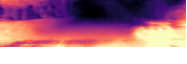





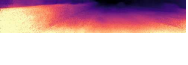






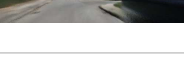




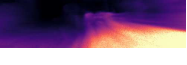








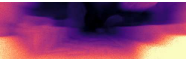
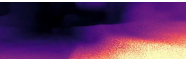









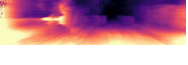








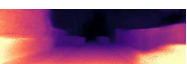
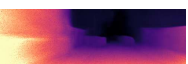
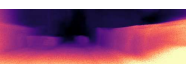

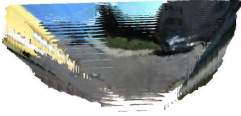
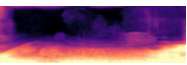
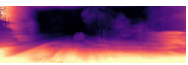
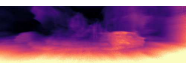


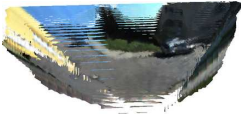
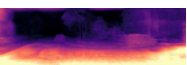
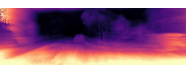
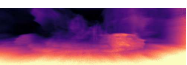


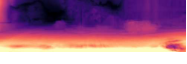
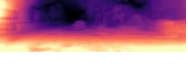
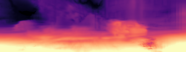


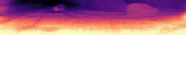
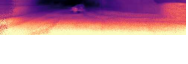
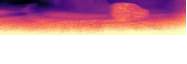

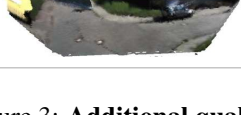



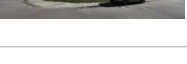
Input	Method	3D mesh	Novel depth			Novel view +5m, +10°
			+1m, 0°	+3m, -10°	+5m, +10°	
	PixelNeRF					
	MINE					
	VisionNeRF					
	<b>SceneRF</b>					
	PixelNeRF					
	MINE					
	VisionNeRF					
	<b>SceneRF</b>					
	PixelNeRF					
	MINE					
	VisionNeRF					
	<b>SceneRF</b>					
	PixelNeRF					
	MINE					
	VisionNeRF					
	<b>SceneRF</b>					

Figure 3: Additional qualitative results on SemanticKITTI [2] (val.).

Input	Method	3D mesh	Novel depth			Novel view +5m, +10°
			+1m, 0°	+3m, -10°	+5m, +10°	
	PixelNeRF					
	MINE					
	VisionNeRF					
	<b>SceneRF</b>					
	PixelNeRF					
	MINE					
	VisionNeRF					
	<b>SceneRF</b>					
	PixelNeRF					
	MINE					
	VisionNeRF					
	<b>SceneRF</b>					
	PixelNeRF					
	MINE					
	VisionNeRF					
	<b>SceneRF</b>					

Figure 4: Additional qualitative results on SemanticKITTI [2] (val.).



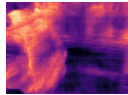
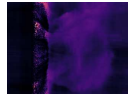




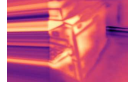
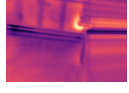


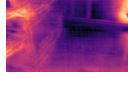




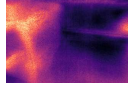
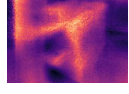
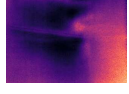

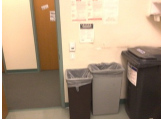

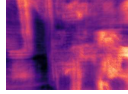

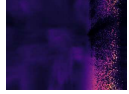







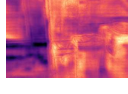
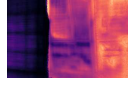
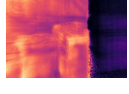


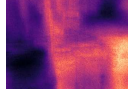
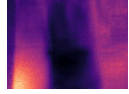
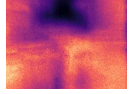



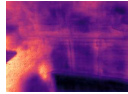
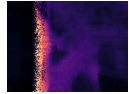
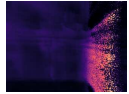



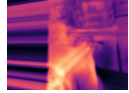
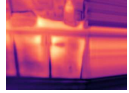


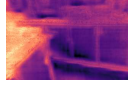
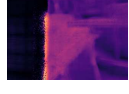



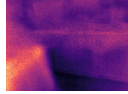
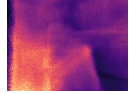
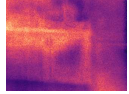
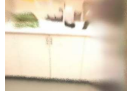
Input	Method	3D mesh (w/ our recons. scheme)	Novel depth			Novel view +0.4m, +20°
			+0.2m, 0°	+0.2m, -20°	+0.4m, +20°	
	PixelNeRF					
	MINE					
	VisionNeRF					
	<b>SceneRF</b>					
	PixelNeRF					
	MINE					
	VisionNeRF					
	<b>SceneRF</b>					
	PixelNeRF					
	MINE					
	VisionNeRF					
	<b>SceneRF</b>					

Figure 5: Additional qualitative results on BundleFusion [7] (val.).



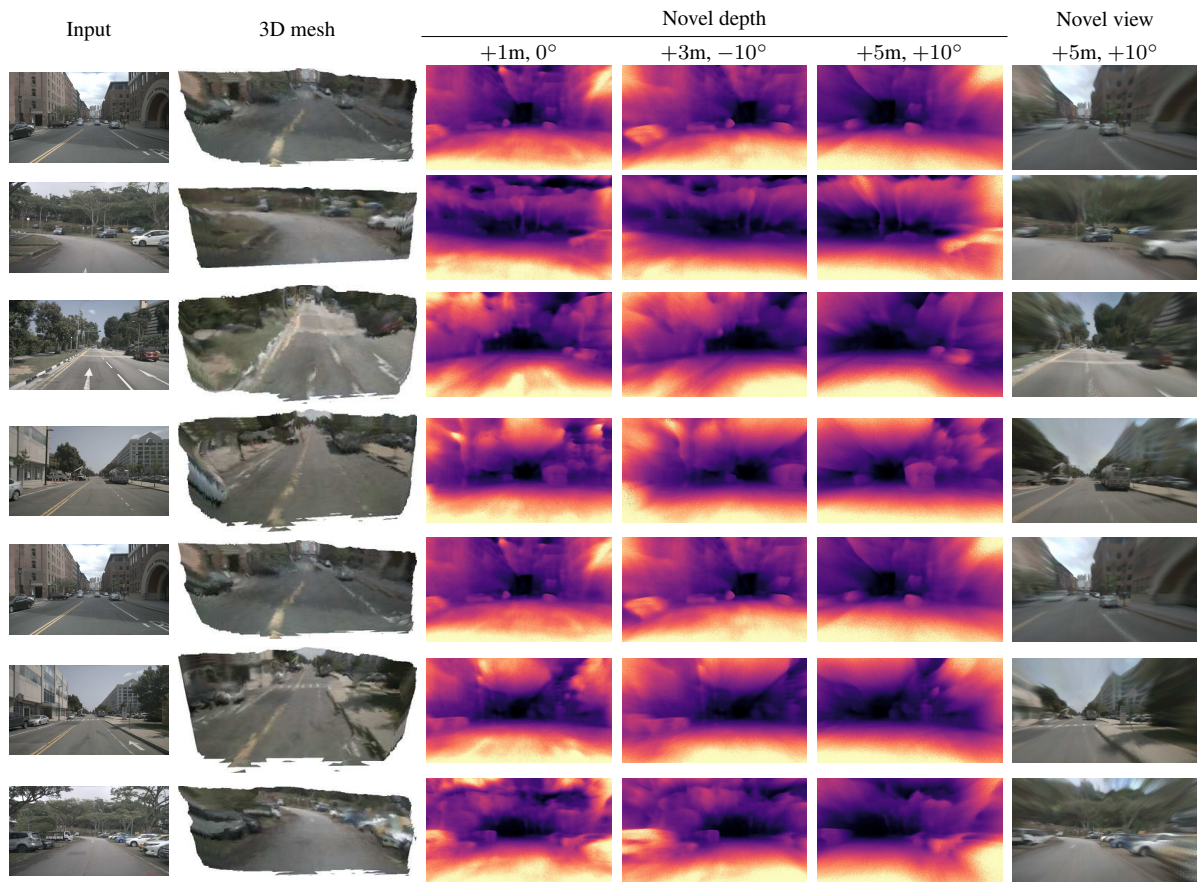


Figure 6: **Additional generalization results on nuScenes [4].** The model is trained only on SemanticKITTI [2].