NEAT: Distilling 3D Wireframes from Neural Attraction Fields

Supplementary Material

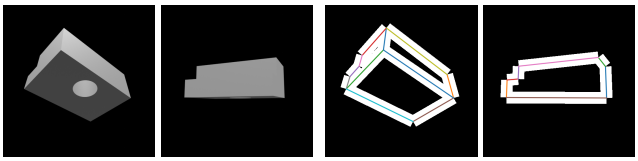
The supplementary document is summarized as follows:

- Appx. A gives a summary of the supplementary video.
- Appx. B elaborates on the technical details (*introduced in Sec. 3.2 of the main paper*) of NEAT optimization.
- Appx. C supplies the details for the final step of distillation for 3D wireframe reconstruction (*introduced in Sec. 3.3* of the main paper).
- Appx. D presents the additional experiments on the ABC dataset [14] to discuss the performance given the ground-truth annotations of 3D wireframes.
- Appx. E quantitatively reports the potential of NEAT for view synthesis with 3D Gaussian Splatting on the DTU dataset.
- Appx. F shows the miscellaneous stuff.

A. Video

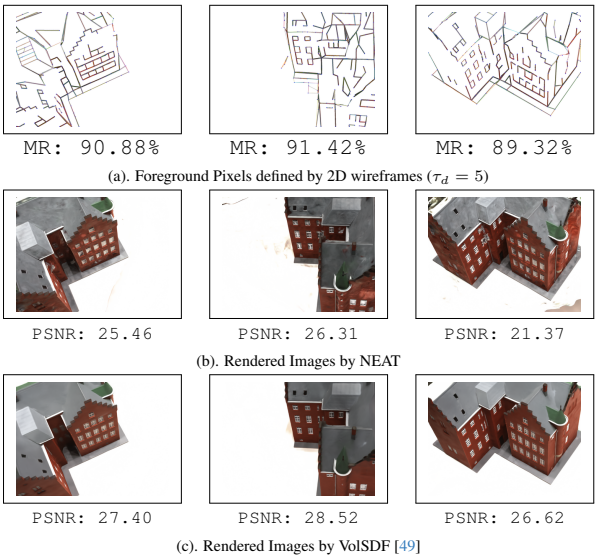
In our supplementary video, we begin by demonstrating the core concepts of our research. Using a basic object from the ABC dataset as an illustrative example, we showcase the 3D line segments learned through the NEAT field, the functionality of the global junction perceiving module, and the construction of the final 3D wireframe model. Following this, the video highlights the learning of redundant 3D line segments and the optimization process for global junctions, using the DTU-24 dataset as a case study. The video concludes with qualitative evaluations on both the DTU and BlendedMVS datasets, providing visual support to the quantitative analyses of the main paper.

B. Optimization of NEAT



(a) Input images (b) 2D Wireframes & rendering pixels

Figure 10. A toy example on the ABC dataset [14] for the foreground pixels defined by the detected 2D wireframes.



(a). Foreground Pixels defined by 2D wireframes ($\tau_d = 5$)

(b). Rendered Images by NEAT

(c). Rendered Images by VolSDF [49]

Figure 11. A comparison for volumetric rendering learned from wireframe-related rays (pixels) *vs*. the vanilla ray sampling. In (a), we show the 2D line segments detected by HAWPv3 [46] and the used foreground pixels in each view. "MR" denotes the mask ratio (the number of foreground pixels among all the pixels). In (b), we show the corresponding views rendered by NEAT that are learned by the foreground pixels in (a). In the bottom (c), we show the rendered images by VolSDF [49] as the reference. In (b) and (c), the PSNR values are marked at the bottom for each view.

B.1. Details on Line Segment Rendering

Our method renders 3D line segments based on the detected 2D wireframes in each view, distinguishing itself from conventional volume rendering approaches that utilize all pixels (rays) for rendering. As demonstrated in Fig. 10 with a toy example from the ABC dataset, only pixels with "white" colors are engaged in the rendering process of 3D line segments. This technique is inspired by the attraction field representations [25, 43–46], where the involved pixels are determined by the perpendicular distance between a point and a line segment. We set a threshold, τ_{ray} (as mentioned in Sec. 3.1 of our main paper), to differentiate the rendering pixels as foreground while disregarding the non-rendering pixels as background. Practically, τ_{ray} is usually set to 5 for training/optimization, and reduced to 1 to minimize computational costs. We refer to this approach as *wireframe-driven ray sampling*.

To demonstrate the effectiveness of wireframe-driven ray sampling, we conducted a series of experiments on `scene 24 from the DTU dataset` [1]. Fig. 11 illustrates

Table 3. The influence of wireframe reconstruction results from different distance thresholds. The larger τ_d value is, the more line segments are involved in the optimization/learning.

	ACC-J↓	ACC-L↓	COMP-L↓	#Lines	#Junctions	MR	PSNR
$\tau_d = 1$	0.853	0.764	6.137	785	540	97.49%	17.79
$\tau_d = 5$	0.639	0.594	5.910	860	528	89.70%	21.55
$\tau_d = 20$	0.578	0.596	6.158	694	508	66.10%	24.68

the feasibility of optimizing coordinate MLPs using this sampling technique. As depicted in Fig. 11(a), by masking over 80% of the pixels (using a distance threshold of 5 pixels), we can still effectively optimize coordinate MLPs, leading to the reasonable outcomes shown in Fig. 11(b).

In addition to rendering results, we observed that increasing the distance threshold leads to a reduction in the number of line segments and junctions. As detailed in Tab. 3, setting the distance threshold to $\tau_d = 20$ results in fewer 3D lines and junctions. Although the ACC errors are marginally reduced, there is an increase in completeness. Conversely, when the distance threshold τ_d is set to 1, a performance degradation is noted across all metrics due to insufficient supervision signals.

B.2. The Number of Global Junctions

The number of global junctions is determined heuristically to encompass all potential 3D junctions. Based on observations from both the DTU and BlendedMVS datasets, where the detected 2D line segments are in the hundreds, we set the estimated number of 3D junctions to 1024. In Tab. 4, we present experiments conducted on the DTU-24 scene with varying numbers of junctions, denoted as N, to assess performance differences. The results indicate that increasing the number of possible global 3D junctions to a larger value (e.g., $N = 2048$) yields only a marginal increase in the count of learned 3D line segments and junctions in the final wireframe models. Conversely, a smaller N tends to result in incomplete 3D wireframe models.

N	# 2D Juncs.	# 3D Junctions	# 3D Lines	ACC-J	ACC-L	COMP-L
1024 (default)	212 (min)	549	860	0.639	0.549	5.910
$N = 128$	297 (max)	99	93	0.422	0.440	8.541
$N = 512$	258.2 (avg)	397	641	0.526	0.574	6.302
$N = 2048$		624	983	0.656	0.599	5.849

Table 4. The performance influence of wireframe reconstruction from different configuration of the number of 3D junctions during optimization.

B.3. Additional Implementation Details

Network Architecture. The coordinate MLPs used in our NEAT approach are derived from VolSDF [49], which contains three coordinate MLPs for SDF, the radiance field, and the NEAT field. For the MLP of SDF, it contains 8 layers with hidden layers of width 256 and a skip connection from the input to the 4th layer. The radiance field and the NEAT field share the same architecture with 4 layers with hidden layers of width 256 without skip connections. The proposed global junction perceiving (GJP) module contains two hidden layers and one decoding layer as described in the code snippets of Sec. 1 in our main paper.

Hyperparameters. The distance threshold τ_d about the foreground pixel (ray) generation is set to 5 by default.For the number of global junctions (*i.e.*, the size of the latent), we set it to 1024 on the DTU and BlendedMVS datasets. When the scene scale is larger (*e.g.*, a scene from ScanNet mentioned in Fig. 5 of the main paper), the number of global junctions is set to 2048. For DBScan [7], we use the implementation from `sklearn` package, set the epsilon (for the maximum distance between two samples) to 0.01 and the number of samples (in a neighborhood for a point to be considered as a core point) to 2.

C. The Final Distillation Step of NEAT

This section elaborates on the final distillation step required in our NEAT methodology for 3D wireframe reconstruction, with a particular focus on the extensive use of global junctions. We aim to provide a detailed insight into this crucial phase of the NEAT process.

To begin with, let us consider the challenge inherent in the junction-driven finalization of NEAT. As depicted in Fig. 12, using a toy ABC scene as an example, we observe that a considerable number of 3D line segments are rendered and aggregated across different views. Concurrently, 3D junctions are dynamically distilled from the NEAT fields. While a simple approach to combine these 3D junctions with the redundant 3D line segments might seem viable, it is critical to address the potential misalignments between the junctions and line segments. To resolve this issue, we employ a least squares optimization combined with an SDF-based refinement scheme. This approach is designed to precisely adjust the position of 3D junctions, thereby ensuring an accurate and coherent reconstruction of the 3D wireframe.

C.1. Least Square Optimization

To be convenient for readers, we copy Eq. (9) in our main paper to Eq. (10),

$$\mathcal{L}(J) = \sum_{(u,v)} \sum_{i=1}^{T_{u,v}} d_{\text{ang}}(\mathbf{l}^0_{u,v}, \mathbf{l}^i_{u,v})^2 + d_{\text{perp}}(\mathbf{l}^0_{u,v}, \mathbf{l}^i_{u,v})^2, \tag{10}$$

which is the main objective function to adjust the junction positions according to the observation from the op-

Table 5. An Ablation study of the SDF-based 3D Junction Refinement on the DTU dataset for the reconstructed 3D wireframes. ACC-J and ACC-L are the evaluation for junctions and line segments.

Scan	NEAT (Final)				NEAT (w/o Non-Linear Optimization)				NEAT (w/o SDF-based Refinement)			
	ACC-J ↓	ACC-L ↓	#Lines	#Junctions	ACC-J ↓	ACC-L ↓	#Lines	#Junctions	ACC-J ↓	ACC-L ↓	#Lines	#Junctions
Avg.	0.772	0.800	624.2	503.5	1.145	0.872	907.7	589.7	1.275	1.044	729.1	514.3
16	0.826	0.788	729	554	0.834	0.829	852	566	1.190	1.045	751	570
17	0.775	0.670	738	546	0.982	0.765	991	651	1.047	0.836	753	557
18	0.643	0.687	701	596	0.930	0.759	993	689	1.040	0.927	821	609
19	0.699	0.692	809	510	0.956	0.703	994	656	1.051	0.863	714	518
21	0.904	0.692	809	571	0.960	0.725	981	654	1.119	0.848	816	581
22	0.634	0.691	758	596	0.896	0.748	939	684	0.976	0.897	769	603
23	0.588	0.619	771	597	0.840	0.703	933	670	0.926	0.821	774	602
24	0.639	0.594	860	549	0.818	0.620	1008	618	0.872	0.748	866	556
37	1.482	1.086	420	405	1.804	1.477	636	565	2.014	1.860	440	425
40	0.630	1.035	137	469	1.342	0.808	1672	591	1.382	0.983	1241	475
65	0.721	1.035	137	171	1.582	1.178	191	221	1.631	1.340	147	185
105	0.720	1.013	621	478	1.793	1.143	702	511	2.053	1.360	657	490

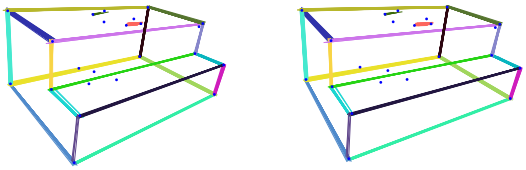


Figure 12. Two different views of the reconstruction of 3D wireframe on the toy scene of ABC dataset before the final distillation step.

timized/learned NEAT field. Here, we mathematically define the alignment cost between the junction-driven 3D line segments $\mathbf{l}^0_{u,v} = (J_u, J_v)$ and its i-th NEAT-field observation $\mathbf{l}^i_{u,v} = (\mathbf{x}^i_u, \mathbf{x}^i_v)$ by the angular cost and the perpendicular cost as follow

$$\begin{aligned} d_{\text{ang}}(\mathbf{l}^0_{u,v}, \mathbf{l}^i_{u,v}) &= 1 - |\langle \frac{J_u - J_v}{\|J_u - J_v\|}, \frac{\mathbf{x}^i_u - \mathbf{x}^i_v}{\|\mathbf{x}^i_u - \mathbf{x}^i_v\|} \rangle|, \\ d_{\text{perp}}(\mathbf{l}^0_{u,v}, \mathbf{l}^i_{u,v}) &= \left\| J_u - \text{proj}(\mathbf{l}^i_{u,v}; J_u) \right\| \\ &\quad + \left\| J_v - \text{proj}(\mathbf{l}^i_{u,v}; J_v) \right\|, \end{aligned} \tag{11}$$

where $\langle \cdot, \cdot \rangle$ is the inner product between two 3D vectors, and the function $\text{proj}(\mathbf{l}^i_{u,v}; J_v)$ projects the point J_v onto the infinite 3D line passing through the line segment $\mathbf{l}^i_{u,v}$. In Tab. 5, we report the performance changes by disabling the non-linear optimization on the DTU dataset, which will result in inferior 3D wireframes with larger ACC errors for both junctions and line segments.

C.2. SDF-based 3D Junction Refinement

Following the non-linear optimization, we employ an SDF-based refinement scheme to further enhance the localization accuracy of junctions. Specifically, for an initial 3D junction $J_i \in \mathbb{R}^3$ and an optimized SDF $d_\Omega(\cdot)$, we refine the location of J_i using the following equation:

$$J^{\text{refined}}_i = J_i - d_\Omega(J_i) \cdot \nabla d_\Omega(J_i), \tag{12}$$

where ∇d_Ω represents the normal direction of the surface at the point J_i.

To assess the impact of this SDF-based refinement on junctions, we conducted an ablation study comparing 3D wireframe models with and without the SDF refinement. The results, presented in Tab. 5, clearly demonstrate the necessity of this refinement step for achieving significantly improved results.

Table 6. The performance change w.r.t. the visibility threshold on the DTU dataset.

Vis	Metric	16	17	18	19	21	22	23	24	37	40	65	105	Avg.
1	ACC.↓	0.788	0.670	0.687	0.692	0.692	0.691	0.619	0.594	1.086	1.035	1.035	1.013	0.800
	COMP.↓	5.414	5.050	5.380	4.653	4.653	5.087	5.599	5.910	7.536	8.783	8.783	6.430	6.106
	Avg. Len.	22.3	23.6	26.7	27.4	27.4	22.8	26.9	27.0	27.9	23.2	23.2	27.5	25.5
	#Lines	729.0	738.0	701.0	809.0	809.0	758.0	771.0	860.0	420.0	137.0	137.0	621.0	624.2
2	ACC.↓	0.770	0.669	0.650	0.642	0.686	0.678	0.604	0.585	1.251	0.755	1.005	1.011	0.776
	COMP.↓	5.493	5.067	5.043	5.562	4.742	5.208	5.670	6.032	7.517	7.027	9.131	6.643	6.095
	Avg. Len.	22.3	23.6	24.4	27.0	27.6	22.8	26.9	27.1	27.4	49.8	22.8	27.0	27.4
	#Lines	711.0	729.0	789.0	667.0	784.0	737.0	756.0	840.0	391.0	1140.0	124.0	572.0	686.7
3	ACC.↓	0.729	0.642	0.640	0.629	0.652	0.639	0.590	0.575	1.188	0.748	0.909	0.981	0.743
	COMP.↓	5.551	5.095	5.117	5.742	4.843	5.357	5.720	6.113	7.473	7.182	9.076	6.785	6.171
	Avg. Len.	22.5	23.7	24.5	27.2	27.8	22.7	26.9	27.2	27.7	49.9	22.8	26.9	27.5
	#Lines	689.0	708.0	765.0	642.0	751.0	708.0	748.0	826.0	371.0	1091.0	112.0	544.0	662.9
4	ACC.↓	0.704	0.619	0.623	0.617	0.607	0.632	0.583	0.556	1.118	0.735	0.891	0.945	0.719
	COMP.↓	5.572	5.256	5.222	5.838	5.021	5.458	5.825	6.168	7.612	7.164	9.220	7.004	6.280
	Avg. Len.	22.5	23.8	24.8	27.5	28.0	22.9	27.0	27.3	27.7	50.5	22.8	26.3	27.6
	#Lines	672.0	679.0	737.0	617.0	723.0	683.0	721.0	806.0	347.0	1052.0	97.0	501.0	636.3

C.3. Visibility Checking

As detailed in Sec. 3.3 of the main paper, we evaluate the reconstructed 3D line segments by projecting them onto 2D images from each view. This process involves computing both the angular and perpendicular distances between the projected 3D line segments and the detected 2D line segments. A 3D line segment is considered to be supported by a 2D detection if it aligns within an angular distance of 10 degrees and a perpendicular distance of 5 pixels, with a minimum overlap ratio of 50%. This

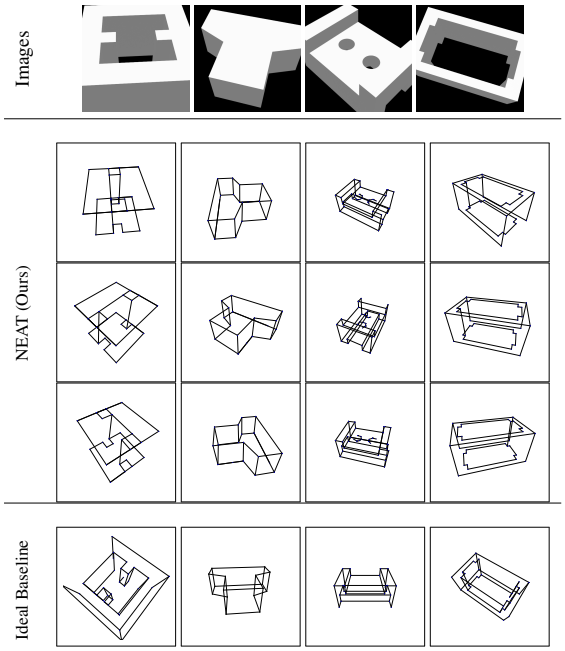


Figure 13. Qualitative Comparisons on ABC objects.

methodology allows us to determine the visibility of each 3D line segment and to filter out those that are invisible as false alarms.

In our standard approach, the visibility threshold for each line segment is set to 1, aiming to achieve a more complete reconstruction. Moreover, we explore the impact of varying this visibility threshold from 1 to 4 on the DTU dataset. The findings, as summarized in Tab. 6, indicate that increasing the visibility threshold results in an improvement in the ACC metric, while the COMP metric increases.

D. Experiments on the ABC Dataset

Because the 3D wireframe annotations are very difficult to obtain for real scene images, to better discuss the problem of 3D wireframe reconstruction and analyze our proposed NEAT approach, we conduct experiments on objects from ABC Datasets as it provides 3D wireframe annotations.

Data Preparation. We use Blender [4] to render 4 objects from the ABC dataset. The object IDs are mentioned in Tab. 7. For each object, we first resize it into a unit cube by dividing the size of the longest side and then moving it to the origin center. Then, we randomly generate 100 camera locations, each of which is distant from the origin by $\sqrt{1.5^2 + 1.5^2} \approx 2.1213$ units. The setting of the distance, $\sqrt{1.5^2 + 1.5^2}$, is from our early-stage development for the rendering, in which we set a camera at $(0, 1.5, 1.5)$ location. By setting the cameras to look at the origin $(0, 0, 0)$, we obtain 100 camera poses. Considering the fact that the ABC dataset is relatively simple, we set the focal length to 60.00mm to ensure the object is slightly occluded for rendering images. The sensor width and height of the camera in Blender are all set to 32mm. The ground truth annotations of the 3D wireframe are from the corresponding `STEP` files. For the simplicity of evaluation, we only keep the straight-line structures and ignore the curvature structures to obtain the ground truth annotations. The rendered images are with the size of 512×512.

Baseline Configuration. Fig. 13 illustrates the rendered input images for the used four objects. Because the rendered images are textureless and with planar objects, the dependency of those baselines on the correspondence-based sparse reconstruction by SfM systems [29] is hardly satisfied to produce reliable line segment matches for 3D line reconstruction. Accordingly, we set up an ideal baseline instead of using Line3D++ [12] and LiMAP [17] for comparison. Specifically, we first detect the 2D wireframes for the rendered input images and then project the junctions and line segments of the ground-truth 3D wireframe models onto the 2D image plane. For the 2D junctions, if a projected ground-truth junction can be supported by a detected one within 5 pixels in any view, we keep the ground-truth junction as the reconstructed one in the ideal case. For the 2D line segments, we compute the minimal value for the distance of the two endpoints of a detected line segment to check if it can support a ground-truth 3D line. The threshold is also set to 5 pixels. Then, we count the number of reconstructed 3D line segments and junctions in such an ideal case.

Evaluation Metrics. For our method, we compute the precision and recall for the reconstructed 3D junctions and line segments under the given thresholds. Because the objects (and the ground-truth wireframes) are normalized in a unit cube, we set the matching thresholds to $\{0.01, 0.02, 0.05\}$ for evaluation. For the matching distance of line segments, we use the maximal value of the matching distance between two endpoints to identify if a line segment is successfully reconstructed under the specific distance threshold. For the ideal baseline, we report the number of ground-truth primitives (junctions or line segments), the number of reconstructed primitives, and the reconstruction rate.

Results and Discussion. Tab. 7 quantitatively summarizes the evaluation results and the statistics on the used scenes. As it is reported, our NEAT approach could accurately reconstruct the wireframes from posed multiview images. The main performance bottleneck of our method comes from the 2D detection results. As shown in the ideal baseline, by projecting the 3D junctions and line segments into the image planes to obtain the ideal 2D detection

ID		Evaluation Results						Ideal Baseline		
		$P_{0.01}$	$P_{0.02}$	$P_{0.05}$	$R_{0.01}$	$R_{0.02}$	$R_{0.05}$	#GT	# Reconstructed	Recon. Rate
4981	J	0.706	0.765	0.882	0.750	0.812	0.938	32	28	0.875
	L	0.758	0.758	0.758	0.521	0.521	0.521	48	41	0.854
13166	J	0.889	0.889	0.889	1.000	1.000	1.000	16	16	1.000
	L	1.000	1.000	1.000	1.000	1.000	1.000	24	24	1.000
17078	J	0.400	0.629	0.686	0.583	0.917	1.000	24	23	0.958
	L	0.408	0.653	0.714	0.556	0.889	0.972	36	32	0.889
19674	J	0.969	1.000	1.000	0.969	1.000	1.000	32	32	1.000
	L	0.969	1.000	1.000	0.969	1.000	1.000	48	40	0.833

Table 7. Evaluation Results and some Statistics on ABC objects. In each object, we evaluate the precision and recall rates for junctions (J) and line segments (L). For the ideal baseline, we count the number of ground-truth primitives, the number of reconstructed 3D primitives, and the reconstruction rate in the ideal baseline.

results, the 2D detection results by HAWPv3 [46] did not perfectly hit all ground-truth annotations. Furthermore, suppose we use the hit (localization error is less than 5 pixels) ground truth for 3D wireframe reconstruction, there is a chance to miss some 3D junctions and more 3D line segments. In this sense, given a relaxed threshold of the reconstruction error for precision and recall computation, our NEAT approach is comparable with the performance of the ideal solution. For the first object (ID 4981), because of the severe self-occlusion, some line segments are not successfully reconstructed for both the ideal baseline and our approach. For object 17078, our NEAT approach reconstructed some parts of the two circles that are excluded from the ground truth, which leads to a relatively low precision rate. Fig. 13 also supported our results.

E. 3D Gaussians with NEAT Junctions

In this section, we extend the application of our NEAT framework to 3D Gaussian Splatting, as proposed by Kerbl et al. [13], by substituting the initial point cloud derived from Structure-from-Motion (SfM) with the junctions identified by NEAT. This experiment is designed to showcase the efficacy of NEAT junctions as a compact initialization method for 3D Gaussian Splatting. Using only a few hundred points, our NEAT junctions demonstrate an enhanced fitting ability on the DTU dataset, as evidenced by improved metrics in both Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM).

The experimental results on 12 scenes from the DTU dataset are detailed in Tab. 8. It is observed that by initializing the 3D Gaussians with NEAT junctions, there is a notable improvement in performance: PSNR increases by 0.38 dB and SSIM improves by 0.0003 points. This finding underscores the effectiveness of NEAT junctions in providing a more precise and compact starting point for 3D Gaussian Splatting.

Table 8. Quantitative comparison between the NEAT junctions and SfM points for the initialization of 3D Gaussian Splatting on the DTU dataset.

Scene ID	NEAT Junctions					SfM Points (by COLMAP [29])				
	PSNR ↑	SSIM ↑	#Points (init)	#Points (7k)	#Points (30k)	PSNR ↑	SSIM ↑	#Points (init)	#Points (7k)	#Points (30k)
DTU-16	**28.7** (+0.7)	**0.889** (+0.006)	554	603k	1,496k	28.0	0.883	22k	558k	1,048k
DTU-17	**29.2** (+0.5)	**0.898** (+0.005)	546	903k	2,279k	28.7	0.893	24k	893k	1,305k
DTU-18	**29.3** (+0.4)	**0.901** (+0.004)	596	629k	1,234k	28.9	0.897	18k	581k	1,078k
DTU-19	**29.6** (+0.4)	0.893 (-0.001)	510	475k	1,140k	29.2	**0.894**	19k	561k	756k
DTU-21	**28.7** (+0.2)	**0.898** (+0.004)	571	725k	1,657k	28.5	0.894	19k	698k	1,528k
DTU-22	**29.1** (+0.2)	**0.892** (+0.005)	596	641k	1,455k	28.9	0.887	21k	615k	1,113k
DTU-23	**28.4** (+0.4)	**0.886** (+0.006)	597	974k	2,243k	28.0	0.880	25k	850k	1,667k
DTU-24	**31.1** (+0.9)	**0.909** (+0.008)	549	587k	1,181k	30.2	0.901	13k	528k	852k
DTU-37	**28.2** (+0.5)	**0.875** (+0.000)	405	420k	1,180k	27.7	**0.875**	27k	409k	713k
DTU-40	**30.6** (+0.2)	**0.862** (+0.002)	422	520k	1,403k	30.4	0.860	32k	515k	1,070k
DTU-65	**32.4** (+0.2)	0.855 (-0.001)	171	139k	294k	32.2	**0.856**	11k	150k	208k
DTU-105	30.8 (-0.1)	0.852 (-0.001)	478	165k	238k	**30.9**	**0.853**	23k	169k	216k
Avg.	**29.68** (+0.38)	**0.884** (+0.003)	499.58	565k	1,317k	29.30	0.881	21k	544k	963k

F. Miscellaneous

F.1. Evaluation Metrics

The Definition of ACC and COMP Metrics. We follow the official evaluation protocol of the DTU dataset [1] to compute the reconstruction accuracy (ACC) and completeness (COMP), which is defined to

$$\text{ACC} = \underset{\mathbf{p}\in P}{\text{mean}} \left(\min_{\mathbf{p}^*\in P^*} \|\mathbf{p} - \mathbf{p}^*\| \right), \tag{13}$$

and

$$\text{COMP} = \underset{\mathbf{p}^*\in P^*}{\text{mean}} \left(\min_{\mathbf{p}\in P} \|\mathbf{p} - \mathbf{p}^*\| \right), \tag{14}$$

where P and P^* are the point clouds sampled from the predictions and the ground truth mesh.

F.2. Information of Used BlendedMVS Scenes

The scene IDs and their MD5 code of the BlendedMVS scenes are:

- Scene-01: `5c34300a73a8df509add216d`
- Scene-02: `5b6e716d67b396324c2d77cb`
- Scene-03: `5b6eff8b67b396324c5b2672`
- Scene-04: `5af28cea59bc705737003253`