HelixSurf: A Robust and Efficient Neural Implicit Surface Learning of Indoor Scenes with Iterative Intertwined Regularization - Supplementary Material

Zhihao Liang[1,*], Zhangjin Huang[1,*], Changxing Ding[1] and Kui Jia[1,2,†]
[1]South China University of Technology, [2]Peng Cheng Laboratory
{eezhihaoliang, eehuangzhangjin}@mail.scut.edu.cn, {chxding, kuijia}@scut.edu.cn,

1. Network Architecture

HelixSurf uses two MLPs to encode the implicit signed distance field (SDF-MLP) and implicit radiance field (RF-MLP), respectively. The architecture of HelixSurf is illustrated in Fig. 1. Notably, SDF-MLP uses Softplus (*i.e.* $\texttt{Softplus}(x) = \frac{1}{\beta} \times \log(1 + \exp(\beta \times x))$) as activation functions, where $\beta = 100$. Specifically, we apply positional encoding $\gamma(\cdot)$ [12] to the input spatial position $\boldsymbol{x}$ as Eq. (1) and apply spherical encoding $\text{Sh}(\cdot)$ [20] to the input view direction $\boldsymbol{v}$.

$$\gamma(\boldsymbol{x}) = (\sin(2^0\pi\boldsymbol{x}), \cos(2^0\pi\boldsymbol{x}), \cdots, \sin(2^{L-1}\pi\boldsymbol{x}), \cos(2^{L-1}\pi\boldsymbol{x})) \quad (1)$$

We set the frequencies L in positional encoding to 6 and set the degrees of spherical encoding to 4.

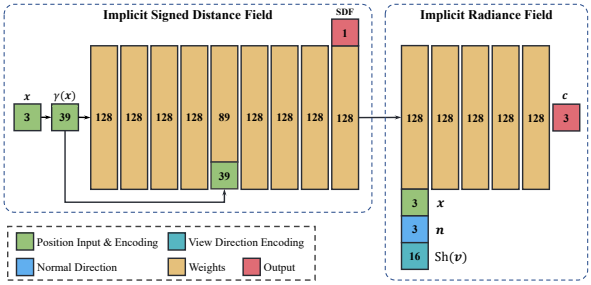


Figure 1. **Architecture of HelixSurf.** HelixSurf takes the position $\boldsymbol{x}$ and the view direction $\boldsymbol{v}$ of the sample point as inputs, and outputs the SDF and color $\boldsymbol{c}$. The surface normal $\boldsymbol{n} = \nabla f(\boldsymbol{x})$.

2. Handling of Textureless Surface Areas

In this work, we treat the areas that the MVS predictions (Fig. 2(a-c)) are less reliable as textureless surface areas (marked as white regions in Fig. 2(e)), and leverage the homogeneity inside individual superpixels to handle these areas. In this scheme, we assume that the superpixels can geometrically partition textureless surface areas. In fact, superpixels (Fig. 2(f)) not only fall in textureless areas but also fall in texture-rich areas or are partially covered by both areas. We thus treat the superpixels (■■■■ in Fig. 2(h)) that are mainly covered by textureless surface areas as the textureless superpixels. Moreover, we treat the superpixel that (Fig. 2(h)) whose smoothness score (*cf*. Algorithm 1) over 0.9 as a textureless superpixel (■ in Fig. 2(h)) to strengthen the regularization. After the identification of textureless surface areas by superpixels, we correspondingly querying the predicted normal map (Fig. 2(i)) from the learned MLP of HelixSurf and denoise the predicted normal with a sliding window manner (Fig. 2(j)).

The primary problem is that the photometric homogeneity inside individual superpixels may not support the correct partition of textureless surface areas at the geometric level (*e.g.*, ■ in Fig. 2(h) confuses the corners). Based on the observation that such superpixels have low smoothness scores (*cf*. Algorithm 1), we thus conduct the adaptive K-means clustering algorithm (*cf*. Algorithm 2) on all textureless superpixels, which adaptively extracts the principal normals for the superpixels. Then, we assign the internal pixels in each textureless superpixels with their corresponding principal normals and obtain the clustered normal map (Fig. 2(k)). We further consider the consistency among multi-view images and conduct mesh-guided consistency on clustered normal maps. Finally, the smooth normal maps (Fig. 2(l)) are used to regularize the learning of the neural implicit surface learning in HelixSurf. The overall textureless surface areas handling scheme is illustrated in Fig. 2.

3. Improving the Efficiency by Establishing Dynamic Space Occupancies

In this work, we devise a scheme that can adaptively guide the point sampling along rays by maintaining dynamic occupancy grids $\mathcal{G}_{\text{Occu}}$ in the 3D scene space.

* indicates equal contribution.
† correspondence to Kui Jia <kuijia@scut.edu.cn>.

For those textureless surface areas totally not covered by the MVS predictions, we initialize them with normals generated with Manhattan assumption [7].

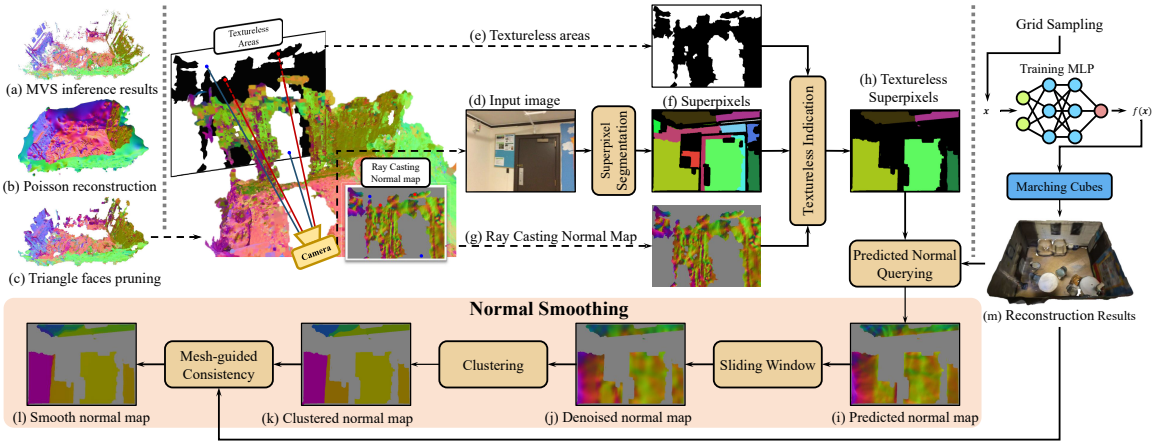


Figure 2. **The overall textureless surface areas handling scheme.**

Algorithm 1 Pseudo code of smoothness scores $\delta(\cdot)$

Input: normals $\{\boldsymbol{n}\}$ for one superpixel $\mathcal{A}$
Initialize: count $= 0$, output$= 0$, $\boldsymbol{n}_{\text{mean}} = \frac{1}{|\{\boldsymbol{n}\}|}\sum \boldsymbol{n}$

```
for n in {n} do
    if n·n_mean / (|n||n_mean|) > 0.9 then
        count = count + 1
    end if
end for
output = count / |{n}|
```

Return: output

Algorithm 2 Pseudo code of adaptive K-means clustering

Input: normals $\{\boldsymbol{n}\}$ for one superpixel $\mathcal{A}$,
smoothness threshold $\tau_{\boldsymbol{n}} = 0.9$,
maximum clustering $k_{\text{max}} = 3$ of `K-means`
Initialize: $k = 1$, output=$\emptyset$

```
while k ≤ k_max do
    {{n}_j}_{j=1}^k = K-means({n}, k)
    if ∀{δ({n}_j) > τ_n}_{j=1}^k then
        for {n_j} in {{n}_j}_{j=1}^k do
            n_principal = 1/|{n_j}| Σ n_j
            push n_principal into output
        end for
        break
    else
        k = k + 1
    end if
end while
```

Return: output

We partition the 3D scene space regularly using a set $\mathcal{G}_{\text{Occu}}$ of occupancy grids of size 64^3, and let the occupancy of any voxel partitioned and indexed by $\{g \subset \mathcal{G}_{\text{Occu}}\}$ be o_g.

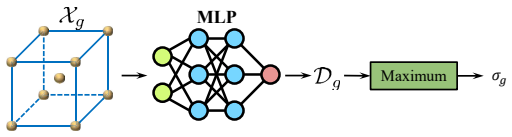


Figure 3. Illustration of calculating the density σ_g of grid g. Note that we follow [16] to model density σ as an SDF-induced volume density.

During training of HelixSurf, we update o_g using exponential moving average (EMA), i.e., $o_g^{\text{EMA}} \leftarrow \max(\sigma_g, \alpha(\sigma_g - o_g^{\text{EMA}}) + o_g^{\text{EMA}})$, where σ_g is the density at g given by the inducing SDF function f and $\alpha = 0.05$ is a decaying factor. We set the voxel indexed by g as occupied if $o_g^{\text{EMA}} > \min(0.01, \text{mean}(\{o_g\}))$. Non-occupied voxels will be skipped directly when performing point sampling along each ray, thus improving the efficiency of differentiable volume rendering used in HelixSurf.

More specifically, given the grid $g \subset \mathcal{G}_{\text{Occu}}$, we update the occupancy o_g of grid g using exponential moving average (EMA), *i.e.*, $o_g^{\text{EMA}} \leftarrow \max(\sigma_g, \alpha(\sigma_g - o_g^{\text{EMA}}) + o_g^{\text{EMA}})$, where σ_g is the density at g by the inducing SDF function f and $\alpha = 0.05$. To calculate the density of g, we set a point set $\mathcal{X}_g = \{\boldsymbol{x}_i^g \in \mathbb{R}^3\}_{i=1}^9$ that contains the center and 8 vertices of this grid. Then, we get the predicted densities $\mathcal{D}_g = \{\sigma_i^g \in \mathbb{R}^+\}_{i=1}^9$ of these points and take the maximum of $\mathcal{D}_g$ as the density of grid σ_g, as illustrated in Fig. 3.

4. More Implementation Details

In this section, we provide more implementation details about the PatchMatch based multi-view stereo (PM-MVS) method, ray casting technique, textureless triangle faces pruning, and the experimental settings.

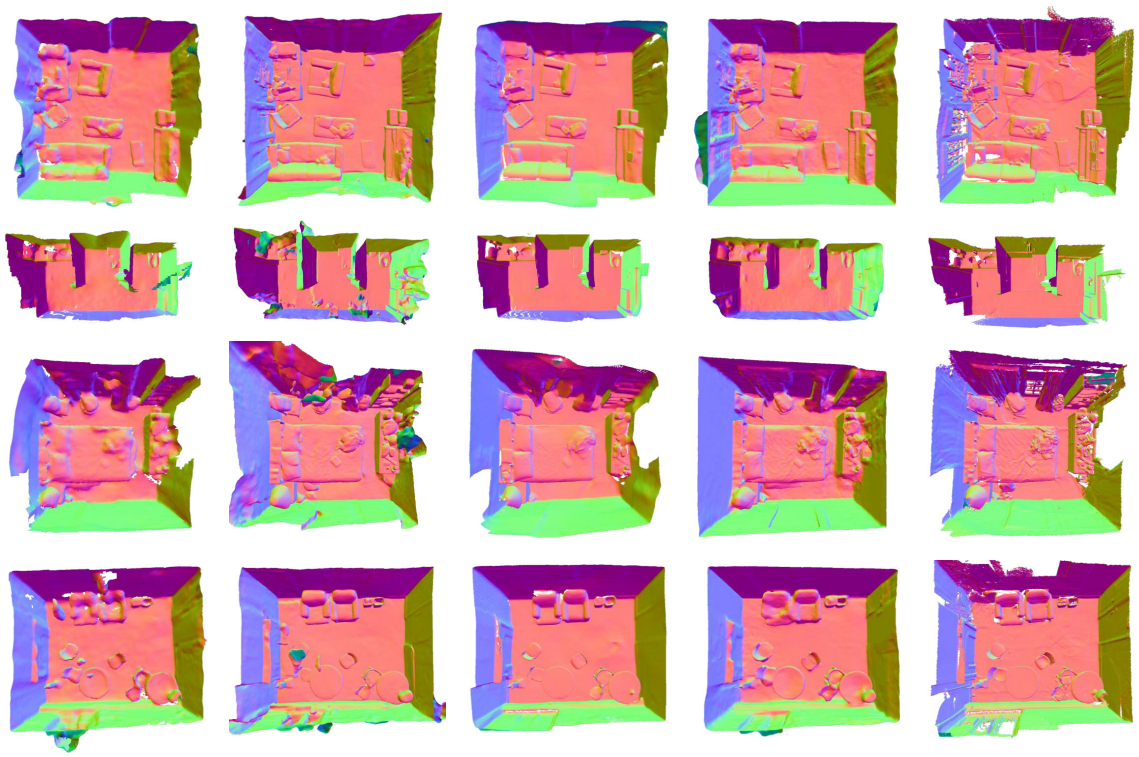


Figure 4. **The top views of the reconstructed scenes on ScanNet.**

4.1. Multi-View Stereo

PM-MVS methods [6, 14, 17, 18, 21] consist of three parts: initialization, iterative sampling & propagation and fusion. The stereo fusion algorithm in COLMAP [14] is time-consuming due to its inefficient interleaved row-/column-wise propagation. To this end, we integrate the ACMH [17] in the HelixSurf. ACMH is also the basis MVS scheme in ACMP [18]. On the one hand, AMCH is based on a checkerboard propagation pattern, which achieves higher parallelization. On the other hand, its proposed adaptively sampling scheme makes the entire mechanism efficient and reliable. While promising, the ordinary ACMH implementation spends a lot of time on I/O operations, and the final fusion step is serially implemented. In this work, we redesign the I/O operations and implement the final fusion step via CUDA kernels.

4.2. Ray Casting

In this work, we apply ray casting to query normal maps on the reconstructed mesh. Intuitively, it's a resource-consuming process since hundreds of thousands of rays need to be cast for each map size of 640×480. For efficiency, we use NVIDIA OptiX [13] technique, a general-purpose ray tracing engine that combines a programmable ray tracing pipeline with a lightweight scene representation. This technique enables us to customize a parallel ray casting program and render hundreds of normal maps in just seconds with an NVIDIA RTX 3090 GPU.

4.3. Textureless Triangle Faces Pruning

For handling textureless surface areas, we utilize the inference results of the integrated PM-MVS method to identify textureless surface areas. Even MVS methods can apply some continuous fitting method (*e.g.*, Poisson reconstruction [9, 10]) to recover a complete surface (*i.e.*, a set of triangle faces) from their inference results (*i.e.*, discrete points), they fail to recover the correct surface of textureless areas due to the lack of inference results on these areas. As investigated in [8], the reconstructed surfaces produce a convex hull or concave envelope results in points missing regions and reconstruct isolated components for noises and outliers. We thus calculate the distance of each triangle face to the nearest point from the inference results, and prune the triangle faces away from the inference results. Then, we remove the isolated components whose diameter is smaller than a specified constant. After pruning the textureless triangle faces, the pruned mesh is used to handle the textureless areas.

4.4. More Experimental Settings

For each scene of ScanNet [3], we uniformly sample one-tenth of views from the frames of the corresponding video, obtain about 200∼500 images and resize them to the size of 640×480 resolution. For Tanks and Temples [11], we use all the images from the provided images set, and resize the images to the size of 960×540 resolution. For both datasets, we follow MVSNet [19] to choose the neighbor referencing images for each view.

5. Evaluation Metrics

In this work, we use the following metrics to evaluate the reconstruction quality: *Accuracy*, *Completeness*, *Precision*, *Recall*, and *F-score*. The definitions of these metrics are shown in Tab. 1. And the metrics for evaluating depth and normal map are shown in Tab. 2.

Metric	Definition
Accuracy	$\text{mean}_{p \in P}(\min_{p^* \in P^*} \|p - p^*\|)$
Completeness	$\text{mean}_{p^* \in P^*}(\min_{p \in P} \|p - p^*\|)$
Precision	$\text{mean}_{p \in P}(\min_{p^* \in P^*} \|p - p^*\| < 0.05)$
Recall	$\text{mean}_{p^* \in P^*}(\min_{p \in P} \|p - p^*\| < 0.05)$
F-score	$\frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$

Table 1. **Evaluation metrics for reconstruction quality used in this work**. P and P^* are the points sampled from the predicted and the ground truth mesh.

	Metric	Definition		Metric	Definition
Depth map	Abs Diff	$\frac{1}{n}\sum \lvert d - d^* \rvert$	Normal map	Mean	$\frac{1}{n}\sum \cos^{-1}[\frac{\lvert \boldsymbol{n} \cdot \boldsymbol{n}^* \rvert}{\lvert \boldsymbol{n} \rvert \lvert \boldsymbol{n}^* \rvert}]$
	Abs Rel	$\frac{1}{n}\sum \frac{\lvert d - d^* \rvert}{d^*}$		Median	$\text{median}\left\{\cos^{-1}[\frac{\lvert \boldsymbol{n} \cdot \boldsymbol{n}^* \rvert}{\lvert \boldsymbol{n} \rvert \lvert \boldsymbol{n}^* \rvert}]\right\}$
	Sq Rel	$\frac{1}{n}\sum \frac{\lvert d - d^* \rvert^2}{d^*}$		RMSE	$\sqrt{\frac{1}{n}\sum (\cos^{-1}[\frac{\lvert \boldsymbol{n} \cdot \boldsymbol{n}^* \rvert}{\lvert \boldsymbol{n} \rvert \lvert \boldsymbol{n}^* \rvert}])^2}$
	RMSE	$\sqrt{\frac{1}{n}\sum \lvert d - d^* \rvert^2}$		Prop_30°	$\frac{1}{n}\#\left\{\boldsymbol{n}, \boldsymbol{n}^* : \cos^{-1}[\frac{\lvert \boldsymbol{n} \cdot \boldsymbol{n}^* \rvert}{\lvert \boldsymbol{n} \rvert \lvert \boldsymbol{n}^* \rvert}] < 30^\circ\right\}$

Table 2. **Evaluation metrics for depth and normal map used in this work.** n is the number of pixels with valid depth or normal in ground truth (GT) depth map or normal map. d and d^* are the predicted and GT depths. $\boldsymbol{n}$ and $\boldsymbol{n}^*$ are the predicted and GT normals.

6. Additional Results

In this section, We discuss the effect caused by the superpixel segmentation. We also provide more experimental results for the ScanNet dataset [3] and Tanks & Temples [11] dataset. Further, we conduct HelixSurf for object-level and real-world scene reconstruction. More visualization details are shown in the attached video.

6.1. Discussions on Superpixel Segmentation

We show quantitative results of that HelixSurf works with varying qualities of superpixels produced by different methods in Tab. 3. The results verify that HelixSurf works stably with these methods.

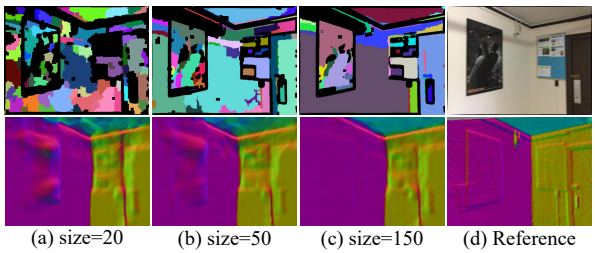
Method	SLIC [2]	Graph-based [5]	SEEDS [4]
F-score↑	0.749	0.755	0.752

Table 3. Results with superpixels produced by different methods. We use a graph-based algorithm [5] to produce superpixels.

We further analyze on varying superpixels sizes. Tab. 4 shows that HelixSurf is robust to different sizes of superpixels. As shown in Fig. 5, our proposed adaptive scheme can tackle low-quality superpixels and produce reliable normals in large textureless surface areas.

Size	20	50	100	150	200
F-score↑	0.738	0.747	0.755	0.756	0.752

Table 4. Results on different sizes of superpixels. `Size` is a hyperparameter in [5] that controls the sizes of produced superpixels. We set `Size` $= 100$ in HelixSurf.



(a) size=20 (b) size=50 (c) size=150 (d) Reference

Figure 5. Results of normal maps produced by different superpixel sizes. The last column presents the input image and ground truth normal map, and the other ones present the results.

6.2. ScanNet

We show more qualitative results in Fig. 4 and Fig. 7. Compared to the state-of-the-art learning-based methods, HelixSurf produces better reconstructions.

6.3. Tanks and Temples

We show more qualitative results on Tanks and Temples [11] in Fig. 8. Our method can produce more precise and complete geometry than baseline methods.

6.4. Object-level Reconstruction

Although our HelixSurf is proposed for scene-level reconstruction, we examine its reconstruction on an object-level dataset (*i.e*. DTU [1]) and report the result in Fig. 10. As can be seen that HelixSurf stands up against baselines.

6.5. Real-World Scene

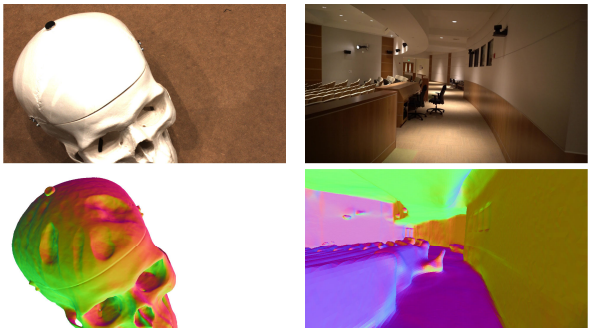
In order to demonstrate the efficacy of HelixSurf in the real-world capture, we conduct HelixSurf on the real-world collected image set (captured by iPhone 11 in [15]). The qualitative reconstruction result is shown in Fig. 11.

7. Novel View Synthesis

In this work, we aim to achieve an accurate and complete reconstruction of the target scene. Furthermore, the accurate reconstruction results enable us to realize high-quality novel view synthesis. For reconstruction, we uniformly sample one-tenth of views from the target scene in ScanNet [3]. For the novel view synthesis, we randomly select some views from the remaining nine-tenths of views and conduct rendering. The rendering results are shown in Fig. 9.

8. Failure Cases

In this work, we assume that textureless surface areas tend to be both homogeneous in color and geometrically smooth. Once the textureless surface areas in the scene do not satisfy this assumption, HelixSurf may fail to handle these areas. For the textureless surface areas with significant curvature as shown in Fig. 6, HelixSurf may fail to handle these textureless surface areas by the normal smoothing scheme and suffer from the artifacts in the reconstruction results. These problems can be solved by adopting some geometric assumptions about the curved surface. However, it will undoubtedly increase the complexity of the whole system. An interesting future work is to introduce more flexible and generalized assumptions to tackle the corner cases.



(a) (b)

Figure 6. **Failure cases.** the first row is the reference images and the second row is the reconstruction results. Surface normals are visualized as coded colors. (a) and (b) show the failure cases at the object-level and the scene-level, respectively.

References

[1] Henrik Aanæs, Rasmus Ramsbøl Jensen, George Vogiatzis, Engin Tola, and Anders Bjorholm Dahl. Large-scale data for multiple-view stereopsis. *International Journal of Computer Vision*, 120(2):153–168, 2016. 5

[2] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282, 2012. 4

[3] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017. 4, 5

[4] Michael Van den Bergh, Xavier Boix, Gemma Roig, and Luc Van Gool. SEEDS: superpixels extracted via energy-driven sampling. *CoRR*, abs/1309.3848, 2013. 4

[5] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International journal of computer vision*, 59(2):167–181, 2004. 4

[6] Silvano Galliani, Katrin Lasinger, and Konrad Schindler. Massively parallel multiview stereopsis by surface normal diffusion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 873–881, 2015. 3

[7] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5511–5520, 2022. 1

[8] Zhangjin Huang, Yuxin Wen, Zihao Wang, Jinjuan Ren, and Kui Jia. Surface reconstruction from point clouds: A survey and a benchmark. *arXiv preprint arXiv:2205.02413*, 2022. 3

[9] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006. 3

[10] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):1–13, 2013. 3

[11] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4):1–13, 2017. 4

[12] 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 *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I*, pages 405–421, 2020. 1

[13] Steven G Parker, James Bigler, Andreas Dietrich, Heiko Friedrich, Jared Hoberock, David Luebke, David McAllis-

ter, Morgan McGuire, Keith Morley, Austin Robison, et al. Optix: a general purpose ray tracing engine. *Acm transactions on graphics (tog)*, 29(4):1–13, 2010. 3

[14] Johannes L Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise view selection for unstructured multi-view stereo. In *European conference on computer vision*, pages 501–518. Springer, 2016. 3

[15] Jiepeng Wang, Peng Wang, Xiaoxiao Long, Christian Theobalt, Taku Komura, Lingjie Liu, and Wenping Wang. Neuris: Neural reconstruction of indoor scenes using normal priors. *arXiv preprint arXiv:2206.13597*, 2022. 5

[16] 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. *Advances in Neural Information Processing Systems*, 34:27171–27183, 2021. 2

[17] Qingshan Xu and Wenbing Tao. Multi-view stereo with asymmetric checkerboard propagation and multi-hypothesis joint view selection. *arXiv preprint arXiv:1805.07920*, 2018. 3

[18] Qingshan Xu and Wenbing Tao. Planar prior assisted patchmatch multi-view stereo. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 12516–12523, 2020. 3

[19] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. *European Conference on Computer Vision (ECCV)*, 2018. 4

[20] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5752–5761, 2021. 1

[21] Enliang Zheng, Enrique Dunn, Vladimir Jojic, and Jan-Michael Frahm. Patchmatch based joint view selection and depthmap estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1510–1517, 2014. 3

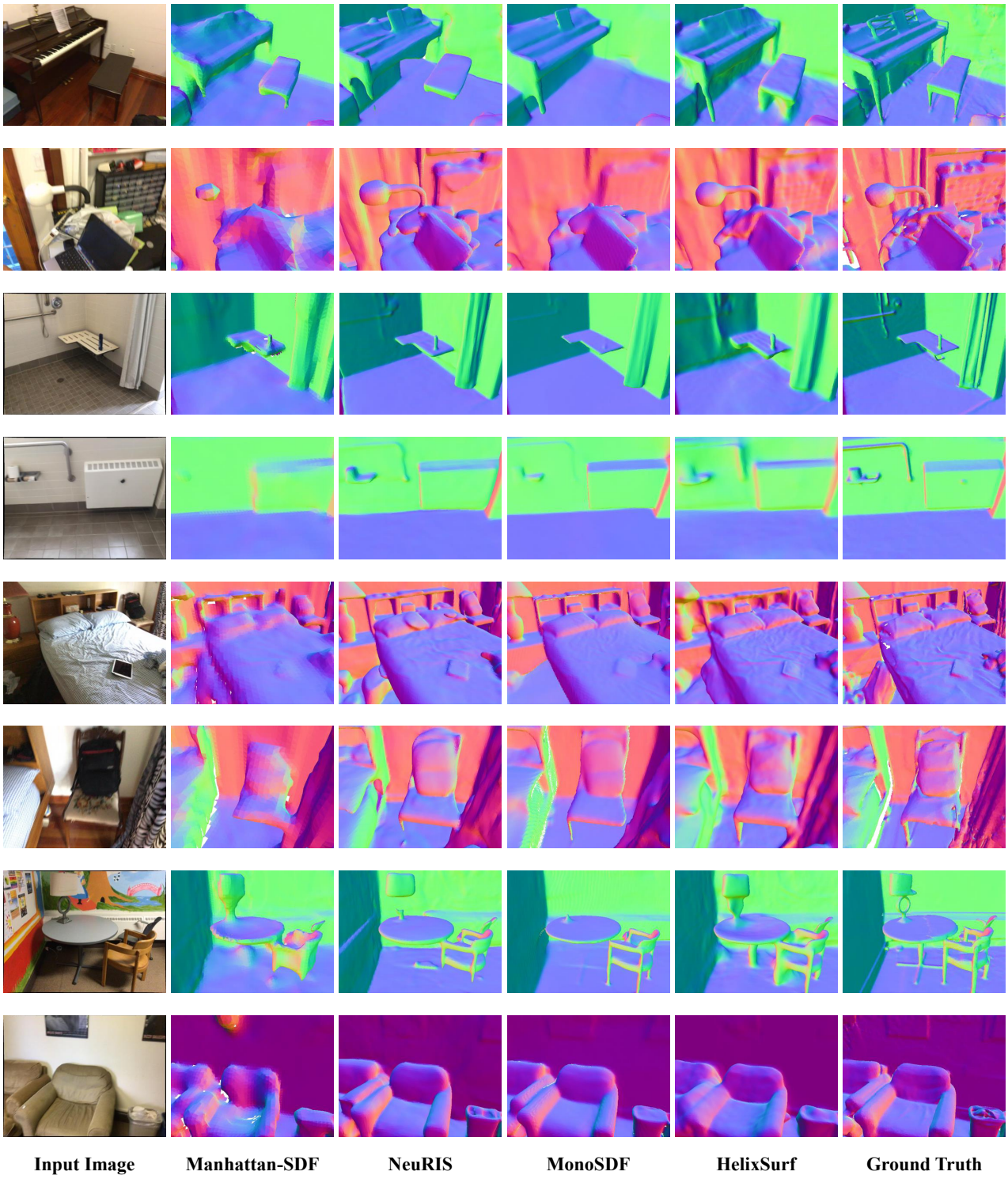


Figure 7. The zoom-in views of the reconstructed scenes on ScanNet.

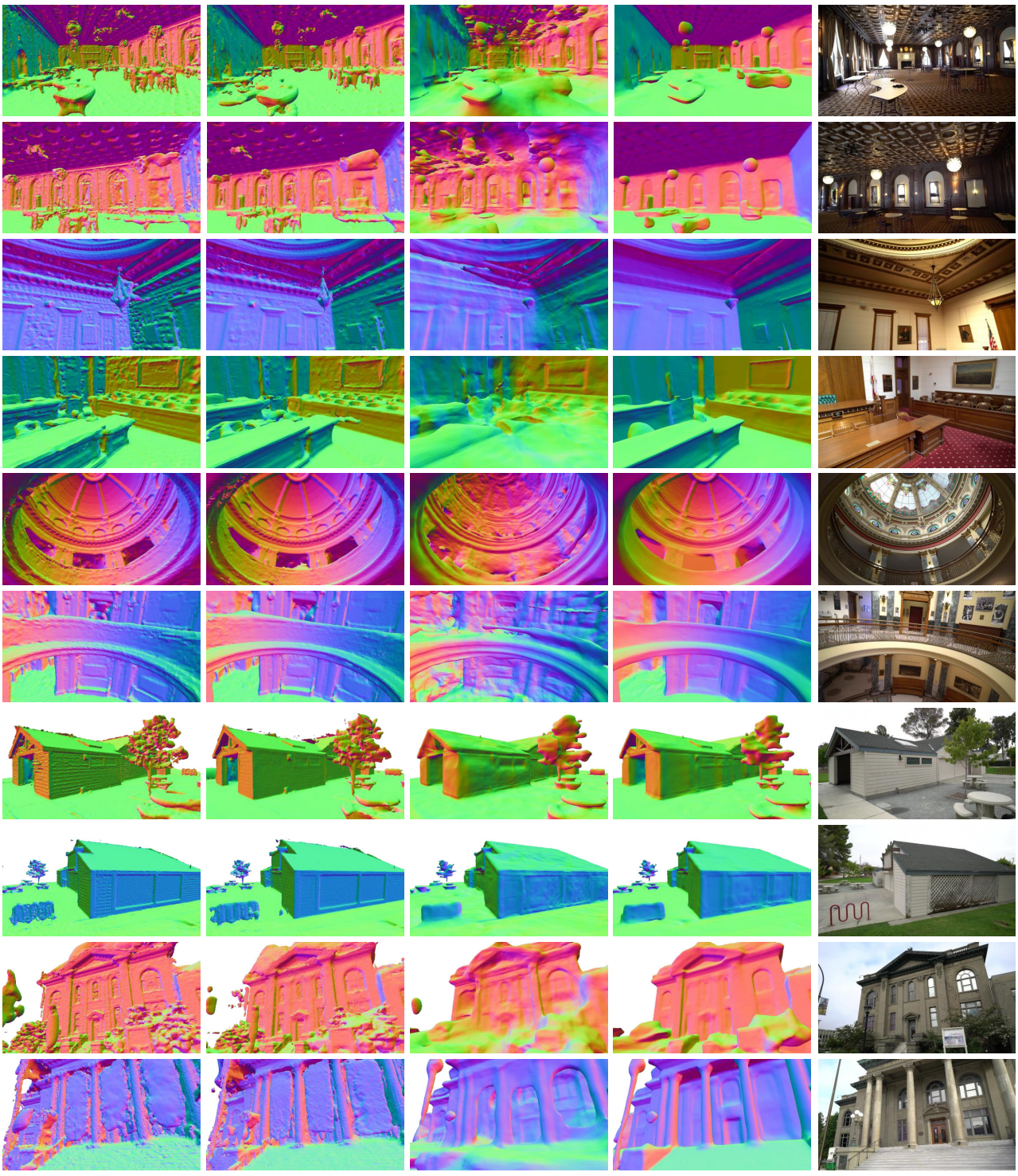


Figure 8. Qualitative Comparison on Tanks and Temples.

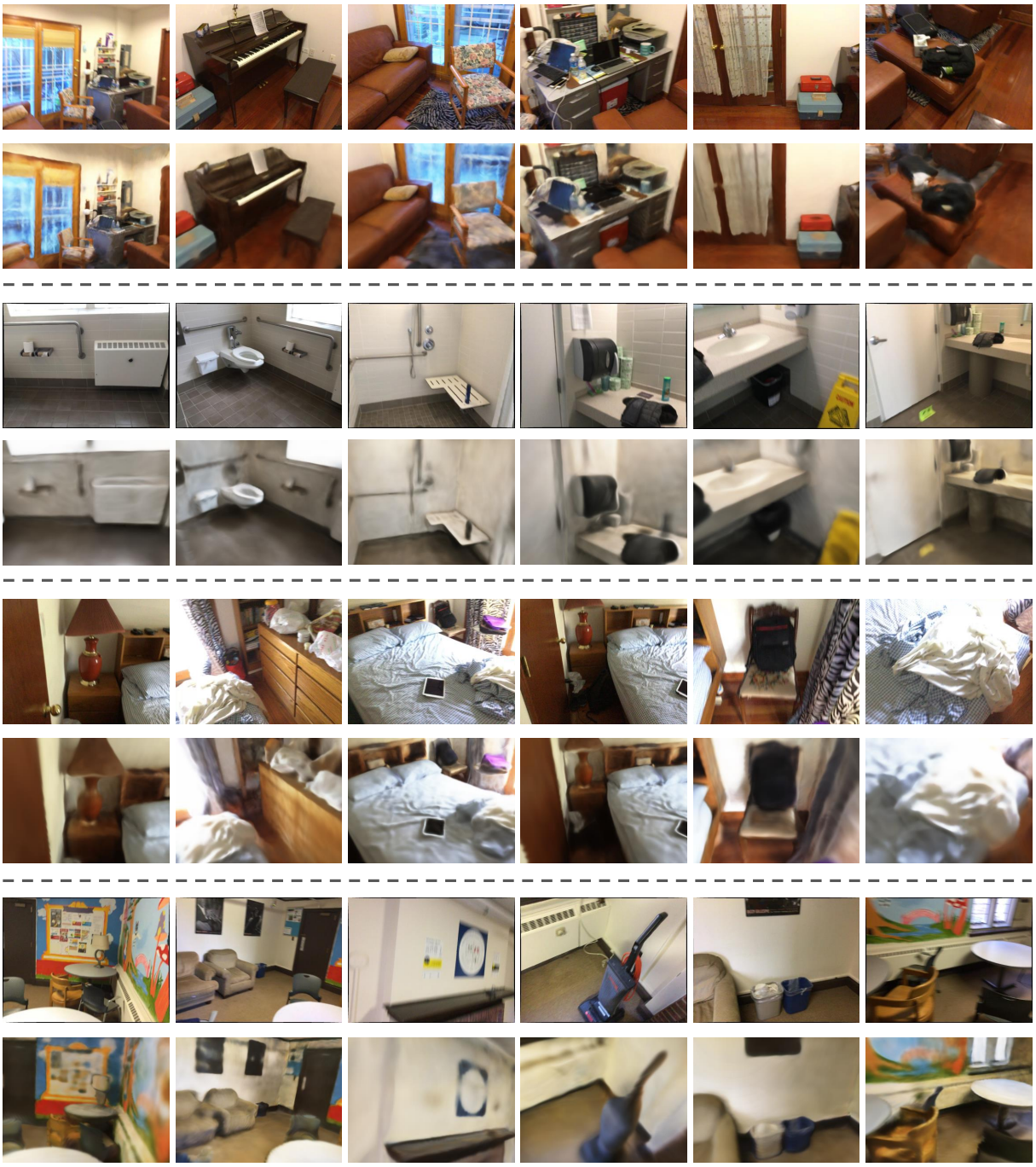


Figure 9. **Novel view synthesis results on ScanNet.** For each block, the first row is the reference images and the second row is the novel view synthesis results.

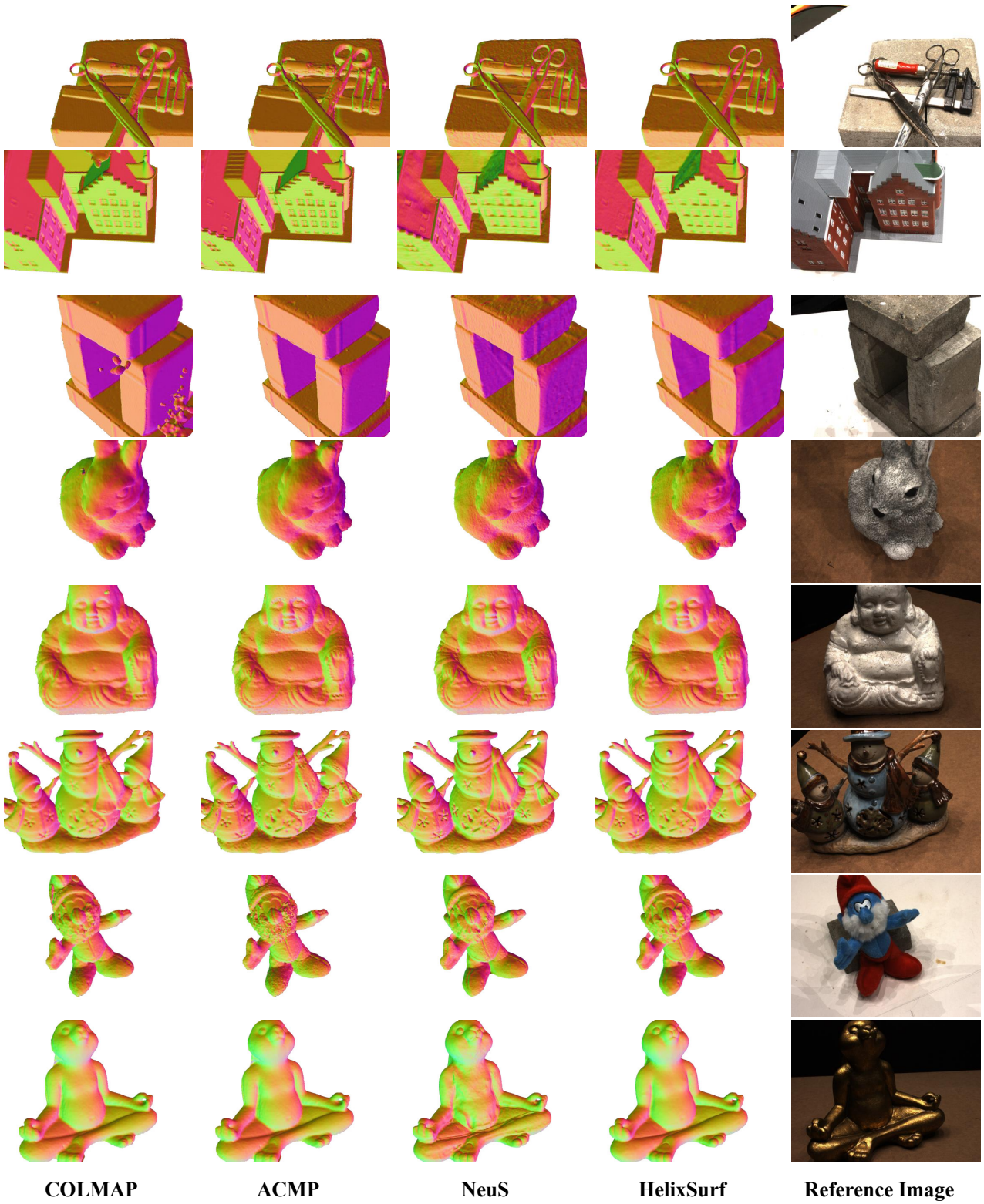
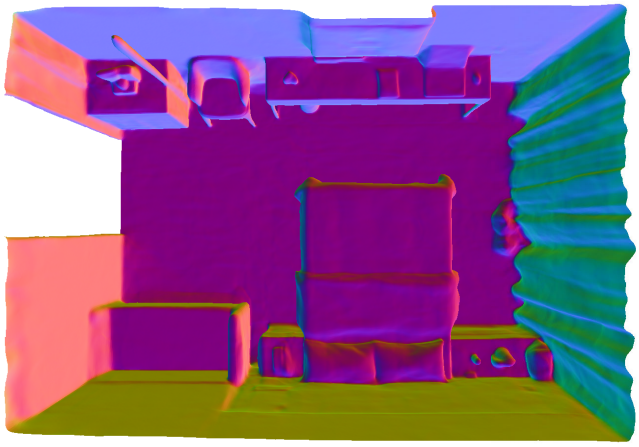
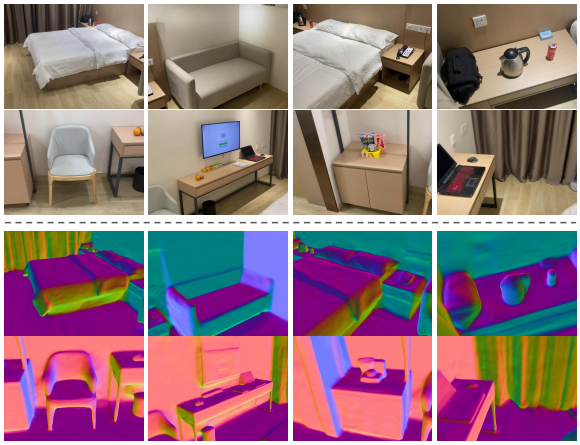


Figure 10. Qualitative Comparison on DTU.



(a) The top view of the entire reconstructed indoor scene.



(b) The first block is a set of reference images. The second block is the corresponding reconstruction results. Surface normals are visualized as coded colors.

Figure 11. **Reconstruction of the real-world capture.**