NeuralUDF: Learning Unsigned Distance Fields for Multi-view Reconstruction of Surfaces with Arbitrary Topologies Supplementary Material

Xiaoxiao Long[1,2†] Cheng Lin[2*] Lingjie Liu[3] Yuan Liu[1] Peng Wang[1]
Christian Theobalt[3] Taku Komura[1] Wenping Wang[4*]
[1]The University of Hong Kong [2]Tencent Games
[3]Max Planck Institute for Informatics [4]Texas A&M University

1. Loss Functions

We train *NeuralUDF* by enforcing the consistency of the rendered colors and the ground truth colors of the input images without using 3D ground truth shapes. Specifically, we optimize our neural networks and two trainable probabilistic parameters κ and β by randomly sampling a batch of pixels and their corresponding rays. Suppose that the size of sampling points in a ray is N and the batch size is M.

The overall loss function is defined as a weighted sum of several loss terms:

$$\mathcal{L} = \mathcal{L}_{\text{color}} + \lambda_0 \mathcal{L}_{\text{patch}} + \lambda_1 \mathcal{L}_{\text{eik}} + \lambda_2 \mathcal{L}_{\text{reg}} \left(+\gamma \mathcal{L}_{\text{mask}}\right). \quad (1)$$

The color loss $\mathcal{L}_{\text{color}}$ is defined as

$$\mathcal{L}_{\text{color}} = \frac{1}{M} \sum_k |\hat{C}_k - C_k|, \quad (2)$$

where $\hat{C}_k$ is the rendered color of the k-th pixel in a batch, and C_k is the corresponding ground truth color.

Same as the prior works [7–9], we utilize the Eikonal term on the sampled points to regularize the UDF field f_u to have unit norm of gradients:

$$\mathcal{L}_{eik} = \frac{1}{NM} \sum_{k,i} \left(\|\nabla f_u\left(\mathbf{r}_k(t_{k,i})\right)\|_2 - 1\right)^2. \quad (3)$$

Optionally, if extra object masks are provided, we can adopt a mask loss:

$$\mathcal{L}_{\text{mask}} = \text{BCE}\left(M_k, \hat{O}_k\right), \quad (4)$$

where M_k is the mask value of the k-th pixel, $\hat{O}_k = \sum_{i=1}^{n} T_{k,i}\sigma_{k,i}\delta_{k,i}$ is the sum of weights of volume rendering along the camera ray, and BCE is the binary cross entropy loss.

*Corresponding authors.
†This work was conducted during an internship at Tencent Games.

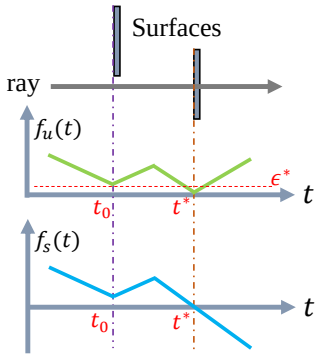

Figure 1. Illustration about zero level sets of UDF and SDF.

Besides minimizing the differences between the rendered color and ground truth color pixel by pixel, we render the colors of a local patch using the patch blending strategy used in [4] and enforce the consistency of the rendered colors and ground truth colors at the patch level by including the loss term:

$$\mathcal{L}_{\text{patch}} = \frac{1}{M} \sum_k \mathcal{R}\left(\hat{P}_k, P_k\right), \quad (5)$$

where the $\hat{P}_k$ are the rendered colors of a patch centering the k-th sampled pixel and the P_k are the corresponding ground truth colors. Empirically, we choose $\mathcal{R}$ to be the Structural Similarity Index Measure (SSIM) metric.

2. Zero Level Set Analysis

The key point of applying volume rendering to UDF is finding the intersection point t^* of the ray and the surface, that is, the zero level set of UDF. However, it's challenging for UDF to find the intersection point t^*, since it's impossible to locate a point with exact zero UDF value. A common practice is to find points whose UDF values are smaller than a pre-defined threshold ϵ. As a result, the approximated

surfaces will be relaxed from exact zero level set to be a ϵ-bounded surface band $\mathcal{S} = \{\mathbf{x} \in \mathbb{R}^3 \mid f_u(\mathbf{x}) < \epsilon\}$.

However, finding an appropriate ϵ is difficult for the reconstruction task. A typical case is shown in Figure 1, a ray first passes by a surface at point t_0 and then hit on a surface at point t^*. It's easy for SDF to locate the intersection point t^* by finding the sign flipping of SDF values. On the contrary, UDF gives all positive values, thus leading to the difficulty in distinguishing the two points t_0 and t^*. To locate the true surface intersection point, the pre-defined ϵ should be smaller than the udf value $f_u(\mathbf{r}(t_0))$, otherwise the point t_0 will be mistakenly located as the intersection point, causing inaccurate volume rendering. Moreover, for objects with complex geometries, a fixed ϵ cannot be adapted to different optimization stages. To tackle these problems, we leverage probabilistic models to design a differentiable indicator function $\Psi(t)$ to locate the surface intersection points.

3. More Implementation Details

3.1. Network architecture

Following the prior works [7–9], we use two MLPs to encode UDF and color respectively. The unsigned distance function f_u is modeled by an MLP that consists of 8 hidden layers with hidden size of 256. A skip connection [6] is used to connect the output of the fourth layer and the input. The color function c is modeled by an MLP with 8 hidden layers with size of 256, which takes the combination of spatial location $\mathbf{r}(t)$, the view direction $\mathbf{v}$, and a 256-dimension feature vector from the UDF MLP as input. Same as the prior works [7–9], we apply positional encoding to the spatial location $\mathbf{r}(t)$ with 6 frequencies and to the view direction $\mathbf{v}$ with 4 frequencies.

3.2. Training details

The Adam [3] optimizer is used to train our networks. We set the learning rate of the UDF MLP as 1e-4 and the other networks and trainable parameters as 5e-4, and the learning rates are controlled by the cosine decay schedule. We set the constant parameter α of the Eq.6 in the main paper as 20, and the constant parameter τ of the Eq.11 in the main paper as 25000. The loss weights of Eq. 1 are set as $\lambda_0 = 0.1, \lambda_1 = 0.1, \lambda_2 = 0.0$ for the DTU [2] dataset, and set as $\lambda_0 = 0.0, \lambda_1 = 0.01, \lambda_2 = 0.01$ for the DeepFashion3D [11] dataset. γ will be set to 0.1 if mask loss term is adopted.

3.3. Hierarchical sampling

To sample more points close to the surfaces, we adopt a hierarchical sampling strategy. We first uniformly sample 64 points along the ray, then we iteratively conduct importance sampling for 5 times. The coarse probability estimation in the i-th iteration is computed by density function with fixed κ and β, which are set as 32×2^i and $32 \times 2^{i+1}$ respectively. We sample extra 16 points in each iteration, and the total number of sampled points is 144. We scale the scene to be reconstructed in the unit sphere, and model the space outside the unit sphere using NeRF++ [10].

4. More Reconstruction Results

Here we show the remaining qualitative comparisons of DeepFashion3D [11] and DTU [2] datasets. Since the DeepFashion3D dataset only provides ground truth point clouds, we use Ball Pivoting meshing algorithm [1] to reconstruct meshes for better visualization. As shown in Figure 4, the SDF based methods model all the shapes as closed surfaces and cannot faithfully recover the captured objects, thus causing erroneous geometries. This is because these methods enforce strong assumption on the closeness of the target shapes. For example, for the first row and the second row, NeuS and VolSDF give closed surfaces for the dress and the trouser where their opening boundaries are even incorrectly connected, thus leading to noticeable errors. For the 8-th row, NeuS attempts to use a double-layered surface to represent the thin cloth; optimizing the tiny SDF band is difficult and its result contains holes. In contrast, our method faithfully reconstruct the geometries with clean and sharp open boundaries.

The remaining comparisons of DTU [2] dataset are plotted in Figure 5 and Figure 6. Unlike the SDF based methods that are particularly designed for closed surfaces, our method adopts UDF as surface representation and does not enforce the closed surface assumption. Although our method doesn't impose the closed surface constraint, our method still achieves comparable performance with the SDF based methods. As shown in the Figure 5, for the second row, NeuS and VolSDF fails to accurately reconstruct the challenging side part of the Brick (the red marked box) since this part is occluded in many input images and lacks enough photography consistency, while our result is more accurate and contain less artifacts. For the 7-th row, the Can with reflecting material, the reconstruction results of NeuS and VolSDF have indentation artifacts on the surface while our method successfully reconstructs the flat and smooth surface.

The experiments validate that, unlike that the SDF based methods are limited to closed shapes, our method is more flexible and can cope with more types of shapes with either open or close surfaces, thus extending the underlying geometric representation of neural volume rendering to various topologies.

5. Novel view synthesis

We conduct the novel view synthesis comparison on the DeepFashion3D dataset. The average PSNR of NeRF, Ours

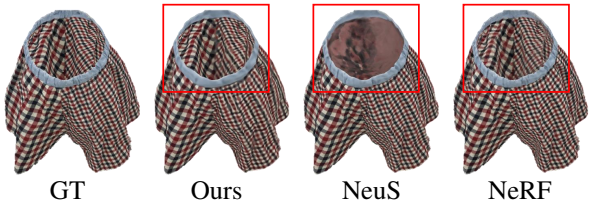

Figure 2. Novel view synthesis.

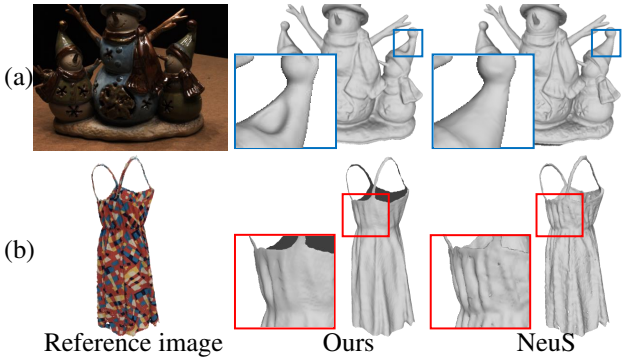

Figure 3. (a) A case of erroneous reconstruction for surfaces with unstable textures. (b) A case that produces smooth surfaces of cloth wrinkles.

and NeuS are 30.69, 29.21, 28.47 respectively. As shown in Figure 2, since NeuS cannot correctly model the geometry of open surfaces, the rendered colors of such regions (red box) have large errors, while ours and NeRF can produce correct colors. We also notice that NeRF, dedicated for novel view synthesis, showing the best quality for this task.

6. Discussions and Limitations

One limitation is that the performance of our method will degrade when the surface textures are unstable and lack enough distinguishable features (textureless or reflective surface). Figure 3 (a) shows a pair of comparison results. With the strong priors of closed and continuous surfaces, SDF can produce reasonable geometry even with unstable textures. Contrarily, UDF does not impose any topology priors and the representation has a much higher degree of freedom. Therefore, when the textures are unstable, the optimization of the accurate surface for UDF can be fairly difficult.

Another limitation is that our method produces smoother results that may lack some geometric details compared with the SDF-based methods. Figure 3 (b) shows a case where our method generates smoother surfaces while does not capture the subtle cloth wrinkles accurately. The leading cause of smoothness is the inherent property of the implicit UDF representation: (1) UDF field does not have sign changes, and thus in practice the neural surface can only be approximated by a ϵ-bounded band $S = \{\mathbf{x} \in \mathbb{R}^3 \mid f_u(\mathbf{x}) < \epsilon\}$; (2) UDF is not differentiable at the zero level sets. To capture more surface details, the ϵ should be minimized in the optimization. However, the non-differentiability means the gradients of the optimized UDF field in the ϵ-bounded band will become increasingly unstable and inaccurate when ϵ is closing to zero. Therefore, optimizing the surface details is more difficult for UDF, compared to SDF that is fully differentiable and has clear sign changes indicating the surfaces.

References

[1] Fausto Bernardini, Joshua Mittleman, Holly Rushmeier, Cláudio Silva, and Gabriel Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE transactions on visualization and computer graphics*, 5(4):349–359, 1999. 2, 4

[2] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 406–413, 2014. 2, 5, 6

[3] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 2

[4] Xiaoxiao Long, Cheng Lin, Peng Wang, Taku Komura, and Wenping Wang. Sparseneus: Fast generalizable neural surface reconstruction from sparse views. *arXiv preprint arXiv:2206.05737*, 2022. 1

[5] 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 *European conference on computer vision*, pages 405–421. Springer, 2020. 5, 6

[6] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019. 2

[7] 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, 2021. 1, 2, 4, 5, 6

[8] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems*, 34, 2021. 1, 2, 4, 5, 6

[9] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *Advances in Neural Information Processing Systems*, 33:2492–2502, 2020. 1, 2, 5, 6

[10] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020. 2

[11] Heming Zhu, Yu Cao, Hang Jin, Weikai Chen, Dong Du, Zhangye Wang, Shuguang Cui, and Xiaoguang Han. Deep fashion3d: A dataset and benchmark for 3d garment reconstruction from single images. In *European Conference on Computer Vision*, pages 512–530. Springer, 2020. 2, 4

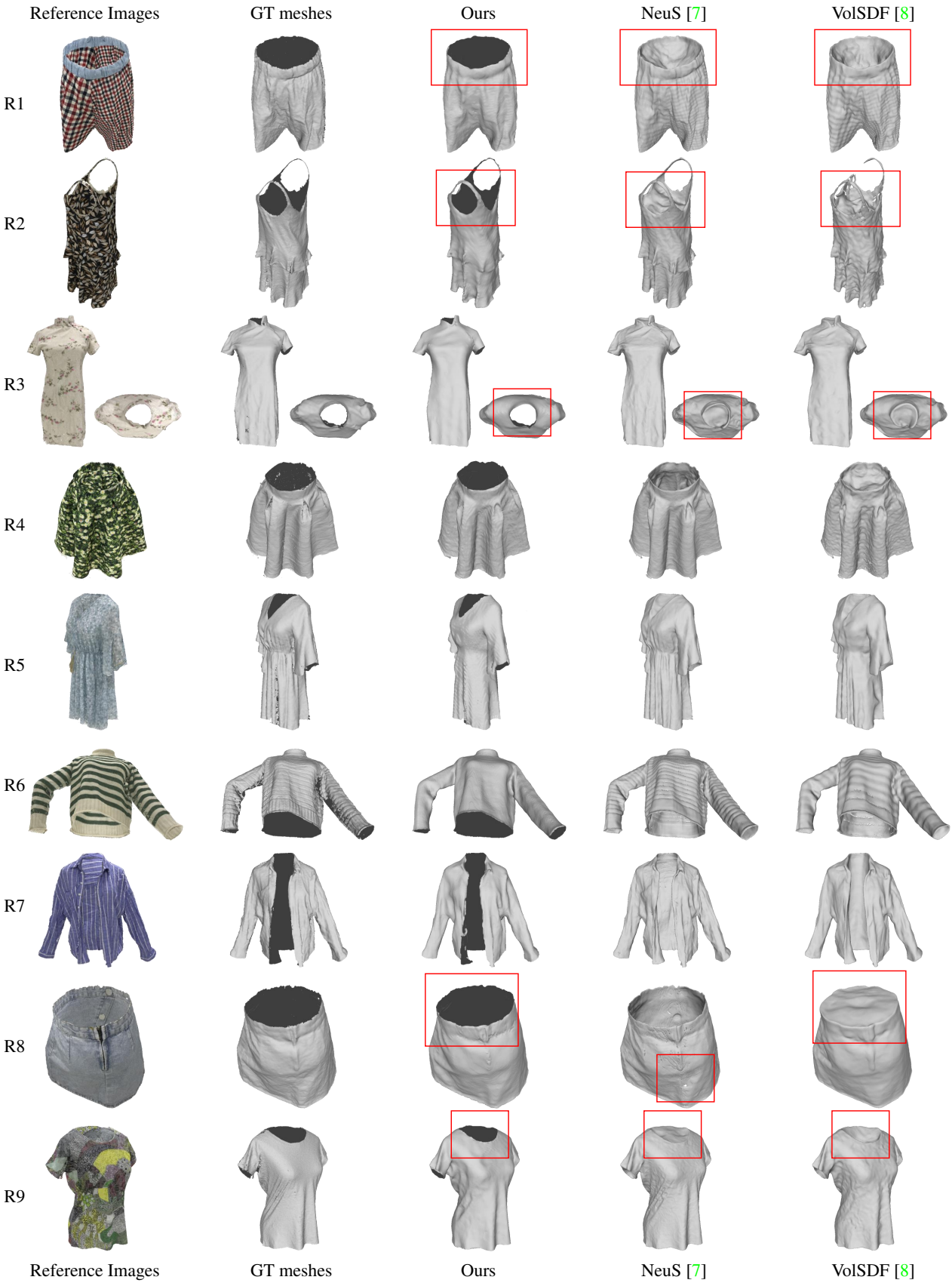

Figure 4. Qualitative comparisons with SOTAs on DeepFashion3D [11] dataset. The GT meshes are reconstructed from GT point clouds provided by DeepFashion3D dataset using Ball Pivoting algorithm [1].

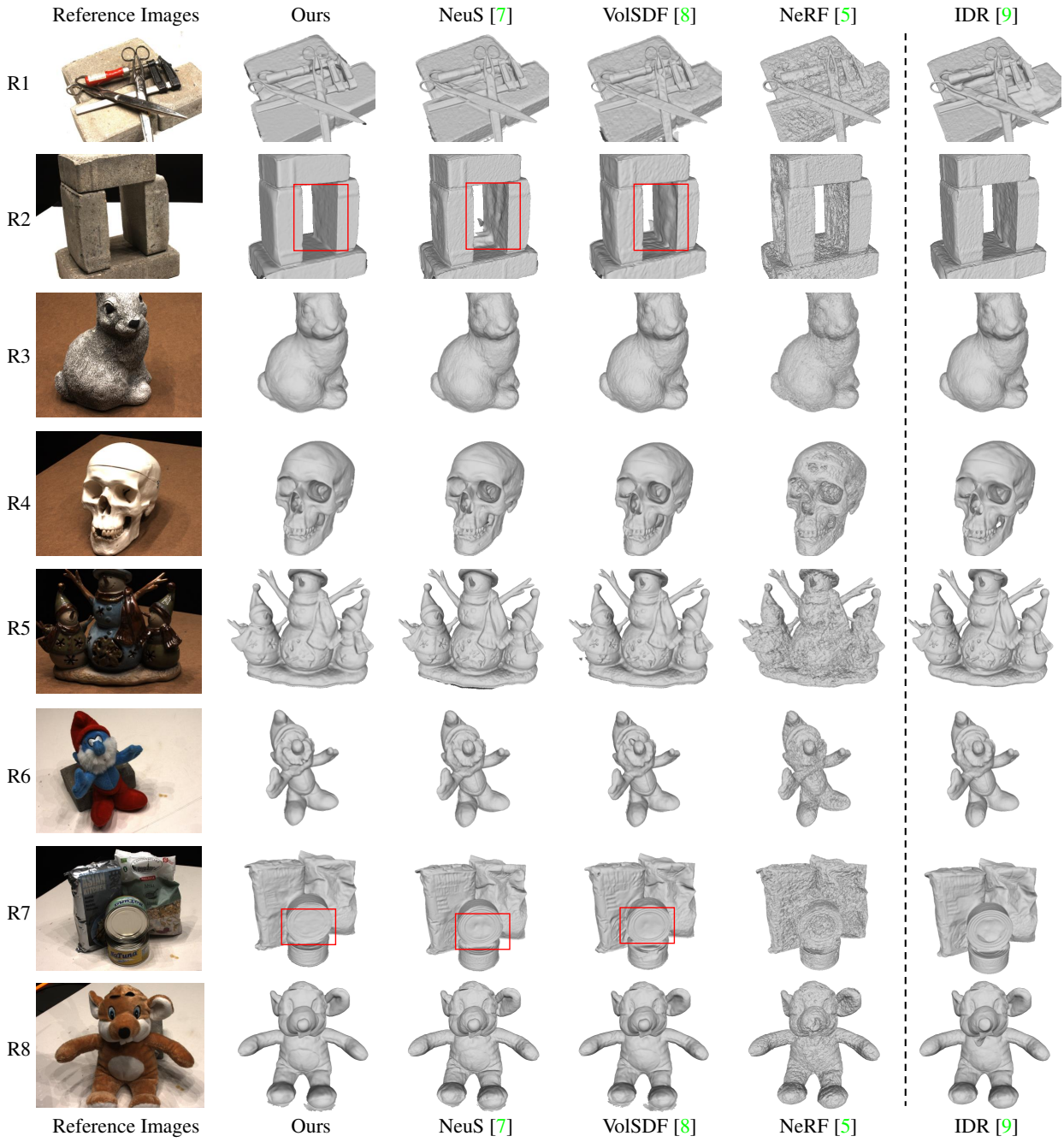

Figure 5. Qualitative comparisons with SOTAs on DTU [2] dataset (Part 1/2).

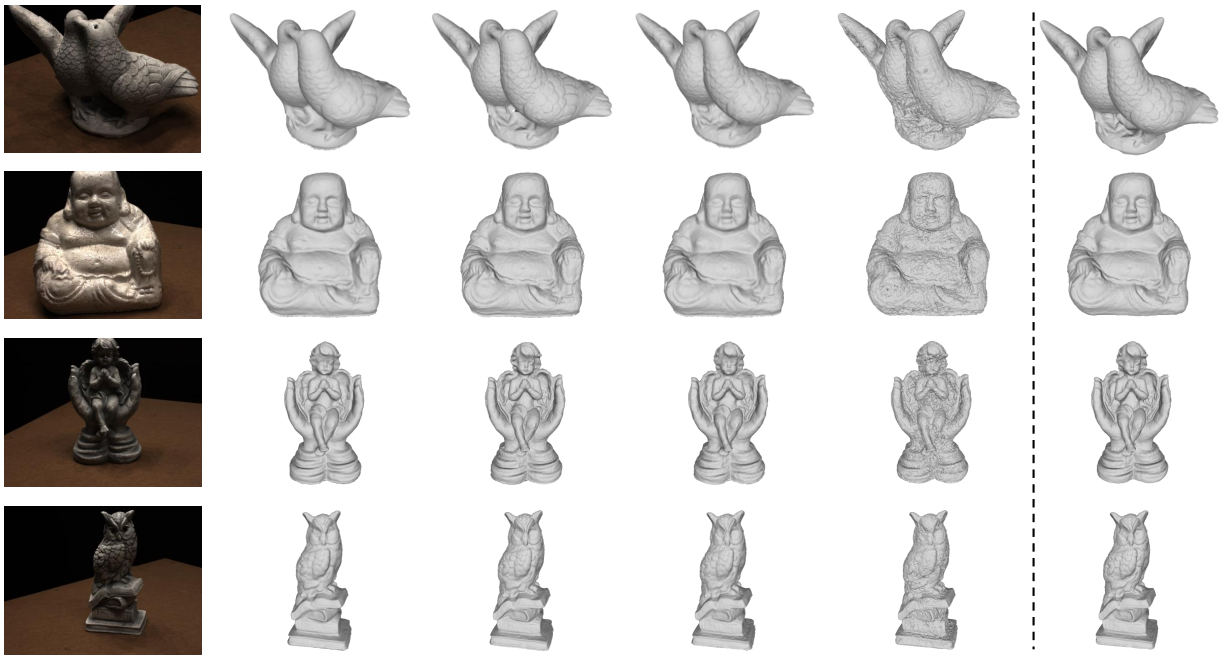

Figure 6. Qualitative comparisons with SOTAs on DTU [2] dataset (Part 2/2).