Gaussian Splatting with Discretized SDF for Relightable Assets

Supplementary Material

1. Model details

In this section, we complete the details of our model, including the losses, hyperparameters, shading model, and lighting model.

1.1. Loss details

We introduce the definition and weights of losses we used in the main paper. The color loss is defined as in 3DGS [4]

$$\mathcal{L}_\mathrm{c} = \lambda ||C_\mathrm{gs} - C_\mathrm{gt}||_1 + (1-\lambda)(1-\mathrm{SSIM}(C_\mathrm{gs}, C_\mathrm{gt})), \quad (1)$$

where C_gs is the rendered color from Gaussians and C_gt is the ground truth color. The coefficient λ is set to 0.8. The supervision L_n from normal is defined as

$$\mathcal{L}_\mathrm{n} = ||\hat{n} - n||^2, \quad (2)$$

where $\hat{n}, n$ are the normal from depth and the normal from Gaussians, respectively. We follow the 2DGS and apply the distortion loss as

$$\mathcal{L}_\mathrm{d} = \sum_{i,j} w_i w_j |D_i - D_j|, \quad (3)$$

where w_i is the blending weight of i-th Gaussian and D_i is the intersection depth for the current pixel. The smoothness loss for BRDF parameters (*i.e.,* albedo **a**, roughness **r**, metallicity **m**) is defined as

$$\mathcal{L}_\mathrm{sm} = ||\nabla p|| \exp^{-||\nabla C_\mathrm{gt}||}, (p \in \{\mathbf{a}, \mathbf{r}, \mathbf{m}\}) \quad (4)$$

where ∇ is the gradient operator and C_gt is the color of ground-truth image. The mask loss is the binary cross entropy between predicted and ground-truth masks. The corresponding weights for the above losses are shown in Tab. 1.

The learning rate of Gaussian attributes follows Gaussian Shader [1], except for the rate of SDF value (set to 0.05). The details of losses are present in the supplementary. The entire training takes about 1 hour on an RTX 4090.

1.2. Hyperparameters

As the median loss is used to promote convergence, we remove this loss when the SDF median $s_\mathrm{m} < 0.2$, which indicates the transformation has been converged. Besides, the relaxing threshold ε for the projection-based consistency loss is set to 0.05. The number N for the spherical initialization is 10^6.

Loss	$\mathcal{L}_\mathrm{n}$	$\mathcal{L}_\mathrm{d}$	$\mathcal{L}_\gamma$	$\mathcal{L}_\mathrm{p}$	$\mathcal{L}_\mathrm{sm}$	$\mathcal{L}_\mathrm{m}$
Weight	0.2	2000	1	10	0.05	0.2

Table 1. The weights of losses present in the paper.

1.3. Split-sum approximation

A typical rendering equation needs to compute an integral on the upper hemisphere involving incident light, view direction, normal, and Bidirectional Reflectance Distribution Function (BRDF):

$$c(\omega_o) = \int_\Omega L_i(\omega_i) f(\omega_i, \omega_o)(\omega_i \cdot n) d\omega_i, \quad (5)$$

where $L_i(\omega_i)$ denotes the light from incident direction ω_i, $f(\omega_i, \omega_o)$ is the BRDF with respect to the incident direction ω_i and outgoing direction ω_o, and n denotes the surface normal. In practice, the Disney Principled BRDF [7] is the most widely used BRDF, consisting of a diffuse lobe and a specular lobe

$$f(\omega_i, \omega_o) = \underbrace{(1-\mathbf{m})\frac{\mathbf{a}}{\pi}}_{\text{diffuse}} + \underbrace{\frac{DFG}{4(\omega_i \cdot n)(\omega_o \cdot n)}}_{\text{specular}}, \quad (6)$$

where D is the normal distribution function, F is the Fresnel term, G is the geometry term, **a** is the albedo, and **m** is the metallicity. However, the integral is expensive to evaluate, and the split-sum technique [3] is an alternative widely used in real-time rendering. The specular term is approximated as

$$c_\mathrm{specular} = L_\mathrm{specular} \cdot ((1-\mathbf{m}) \times 0.04 + \mathbf{m} \times \mathbf{a}) \times F_1 + F_2, \quad (7)$$

where F_1, F_2 are two scalars from a pre-computed table. The diffuse term is computed as

$$c_\mathrm{diffuse} = \frac{\mathbf{a}(1-\mathbf{m})}{\pi} \cdot L_\mathrm{diffuse}. \quad (8)$$

L_diffuse and L_specular are directed queried for a pre-filtered environment map or neural light representations. The rendered color $c(\omega_o)$ is computed as

$$c(\omega_o) = c_\mathrm{diffuse} + c_\mathrm{specular}. \quad (9)$$

1.4. Lighting model

Considering the efficiency of the light query, we use a simple differential environment map and do not model the indirect illumination and occlusion. The environment light is in the cube map format, whose resolution is $6 \times 512 \times 512$.

	MII	TensoIR	TensoSDF	NeRO	Ours
	PSNR / SSIM / LPIPS	PSNR / SSIM / LPIPS	PSNR / SSIM / LPIPS	PSNR / SSIM / LPIPS	PSNR / SSIM / LPIPS
Angel	16.24 / 0.8236 / 0.1404	10.24 / 0.2238 / 0.2739	20.40 / 0.8969 / 0.0871	16.21/ 0.7819 / 0.1923	22.03 / 0.8919 / 0.0819
Bell	17.41 / 0.8594 / 0.1534	10.11 / 0.1018 / 0.2806	29.91 / 0.9767 / 0.0263	31.19 / 0.9794 / 0.0189	24.67 / 0.9280 / 0.0842
Cat	17.68 / 0.8521 / 0.1429	9.10 / 0.1644 / 0.2146	26.12 / 0.9354 / 0.0675	28.42 / 0.9579 / 0.0455	26.48 / 0.9374 / 0.0661
Horse	20.98 / 0.8997 / 0.0713	10.42 / 0.1931 / 0.2913	27.18 / 0.9567 / 0.0318	25.56 / 0.9437 / 0.0410	24.01 / 0.9481 / 0.0351
Luyu	17.89 / 0.8050 / 0.1393	8.27 / 0.2375 / 0.2463	19.91 / 0.8825 / 0.0807	26.22 / 0.9092 / 0.0696	23.80 / 0.9017 / 0.0699
Potion	17.13 / 0.8094 / 0.1747	6.21 / 0.0846 / 0.2954	27.71 / 0.9422 / 0.0759	30.14 / 0.9561 / 0.0623	27.31 / 0.9280 / 0.0982
Tbell	16.54 / 0.8262 / 0.1938	7.47 / 0.1609 / 0.2786	23.33 / 0.9404 / 0.0543	25.45 / 0.9607 / 0.0407	23.66 / 0.9191 / 0.0981
Teapot	16.71 / 0.8546 / 0.1426	9.96 / 0.2093 / 0.2030	25.16 / 0.9482 / 0.0485	29.87 / 0.9755 / 0.0193	24.19 / 0.9293 / 0.0760
Mean	17.57 / 0.8413 / 0.1448	8.97 / 0.1719 / 0.2605	24.97 / 0.9349 / 0.0590	26.63 / 0.9331 / 0.0612	24.52 / 0.9229 / 0.0762
Training	4h	5h	6h	12h	1h
Ren. FPS	1/30	1/60	1/4	1/4	143
Memory	12G	23G	20G	8G	4G

Table 2. The quantitative comparison with NeRF-based methods on the Glossy Blender dataset in terms of PSNR↑, SSIM↑, and LPIPS↓. Numbers in red indicate the best performance, numbers in orange indicate the second best, and numbers in yellow indicate the third best. Although our method has a performance gap compared to NeRO and TensoSDF, our method is more efficient (at most **50% memory usage and 17% training time).**

	MII	NeRO	TensoSDF	TensoIR	Ours
	PSNR / SSIM / LPIPS	PSNR / SSIM / LPIPS	PSNR / SSIM / LPIPS	PSNR / SSIM / LPIPS	PSNR / SSIM / LPIPS
Armad.	26.85 / 0.9441 / 0.0692	23.02 / 0.9335 / 0.0644	23.02 / 0.9355 / 0.0578	34.51 / 0.9754 / 0.0368	31.05 / 0.9621 / 0.0536
Ficus	20.65 / 0.9068 / 0.0728	27.43 / 0.9404 / 0.0677	28.53 / 0.9499 / 0.0533	24.32 / 0.9465 / 0.0543	27.85 / 0.9639 / 0.0390
Hotdog	22.65 / 0.9011 / 0.0893	20.45 / 0.9262 / 0.0888	20.47 / 0.9241 / 0.0906	27.92 / 0.9324 / 0.0833	26.23 / 0.9360 / 0.0746
Lego	23.20 / 0.8643 / 0.1715	17.76 / 0.8577 / 0.1228	17.92 / 0.8670 / 0.1088	27.61 / 0.9253 / 0.0702	25.81 / 0.9087 / 0.0791
Mean	23.34 / 0.9041 / 0.1007	22.17 / 0.9145 / 0.0859	22.48 / 0.9191 / 0.0776	28.59 / 0.9449 / 0.0612	27.78 / 0.9427 / 0.0626

Table 3. The quantitative comparison with NeRF-based methods on the TensoIR Synthetic dataset in terms of PSNR↑, SSIM↑, and LPIPS↓. Numbers in red indicate the best performance, numbers in orange indicate the second best, and numbers in yellow indicate the third best.

2. Proof of the projection-based loss

This part reveals how our projection-based consistency loss approximates the Eikonal loss.

Proposition 1. *Consider a differentiable function f in the region Ω, for $\forall x_1 \in \Omega, f(x_1) \neq 0$ and its projection point $x_0 = x_1 - \nabla f(x_1)/|\nabla f(x_1)| \cdot f(x_1)$, if we guarantee x_0 on the surface ($f(x_0) = 0$), the Eikonal condition $|\nabla f(x)| = 1$ can be approximated at least in a narrow thin-shell space surrounding the surface.*

Proof. The Taylor expansion of f at position x_1 is

$$f(x) = f(x_1) + \nabla f(x_1)(x - x_1) + O((x - x_1)^2) \quad (10)$$

where $\nabla f(x_1)$ means the gradient of f at x_1 and $O((x - x_1)^2)$ means the higher-order term of f. Then substitute x_0 into $f(x)$:

$$\begin{aligned} &x_0 - x_1 = -f(x_1) \cdot \nabla f(x_1)/|\nabla f(x_1)|, \\ &f(x_0) = f(x_1) - f(x_1)\frac{\nabla f(x_1)^2}{|\nabla f(x_1)|} + O((x_0 - x_1)^2) \quad (11) \\ &f(x_1)(1 - |\nabla f(x_1)|) + O((x_0 - x_1)^2) = 0 \end{aligned}$$

which means $|\nabla f(x)| = 1$ is approximated for each x_1 where x_0, x_1 is close enough to make the high-order term $O((x_0 - x_1)^2)$ negligible. Therefore, we can guarantee the Eikonal condition near the surface where Gaussians exist.

	GS-ROR	Ours w/o mask	Ours w/ mask
	PSNR / SSIM	PSNR / SSIM	PSNR / SSIM
Angel	20.81 / 0.8775	21.03 / 0.8842	22.03 / 0.8919
Horse	23.31 / 0.9376	23.35 / 0.9406	24.01 / 0.9481
Teapot	21.17 / 0.8932	22.57 / 0.9087	24.19 / 0.9293

Table 4. The impact of the mask loss on our method.

3. More results

In this section, we present the ablation of detailed choices in our model and more results of our method. Additionally, we select some representative NeRF-based methods for further validation, including MII [10], TensoIR [2], TensoSDF [5], and NeRO [6].

ε	0.01	0.05	0.1	0.2
	PSNR / SSIM	PSNR / SSIM	PSNR / SSIM	PSNR / SSIM
Angel	21.78 / 0.8860	21.91 / 0.8880	22.03 / 0.8919	21.54 / 0.8769
Horse	22.10 / 0.9214	22.51 / 0.9317	24.01 / 0.9481	20.12 / 0.9220
Teapot	23.28 / 0.9190	23.84 / 0.9232	24.19 / 0.9293	23.37 / 0.9200

Table 5. Ablation study on the threshold ε in our method.

Iterations	7K	15K	30K
	PSNR / SSIM	PSNR / SSIM	PSNR / SSIM
No reg.	17.96 / 0.8881	22.13 / 0.9175	22.82 / 0.9330
$\gamma = \gamma_{\mathrm{m}}$	18.85 / 0.9032	22.46 / 0.9239	22.91 / 0.9332
$\mathcal{L}_\gamma$	22.20 / 0.9221	23.64 / 0.9319	24.19 / 0.9481

Table 6. Ablation study on the median loss $\mathcal{L}_\gamma$ in our method. "No reg." means imposing no regularization on the transformation.

	GS-IR	GShader	R3DG	GS-ROR	Ours
	PSNR / CD	PSNR / CD	PSNR / CD	PSNR / CD	PSNR / CD
Ball	18.30/—	30.40/—	21.39/—	35.50/—	34.64/—
Car	25.30/0.61	28.39/0.26	26.59/0.23	30.52/0.17	30.55/0.14
Coffee	30.72/1.22	30.79/1.05	32.57/1.14	30.79/1.51	32.91/0.92
Helmet	25.08/1.22	26.95/1.09	28.78/1.31	32.62/0.23	30.06/0.15
Toaster	18.66/0.72	23.95/0.75	20.07/0.80	25.89/0.53	25.02/0.68
Teapot	38.21/—	43.35/—	43.86/—	43.88/—	43.39/—

Table 7. Comparison in terms of PSNR and CD↓ $(\times 100)$.

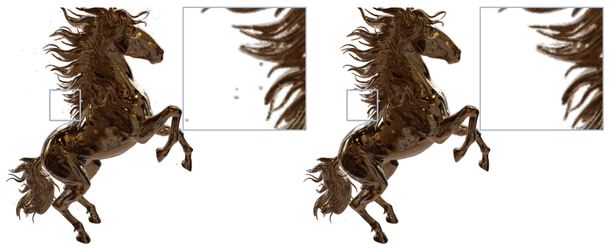

Figure 1. Without the projection-based consistency loss, we observe the artifacts caused by outliers (left). The loss encourages the outlier towards the surface, thus diminishing the artifacts.

3.1. Extra ablation studies

We ablate our hyperparameter choice and loss usage in this part, including the threshold in the projection-based loss, the impact of the median and the mask loss.

The choice of the threshold ε. A large threshold forces Gaussians onto wrong surfaces, while a small one causes most Gaussians to be excluded, resulting in decreased performance. Therefore, we choose to set $\varepsilon = 0.1$ for all scenes, which reveals its generalization ability and yields the best performance in Tab. 5.

The impact of the median loss. We use the median loss to encourage γ towards γ_{m}, instead of directly setting $\gamma = \gamma_{\mathrm{m}}$, because γ_{m} is used to give the direction of optimization and not the target. Besides, the median loss enables the further optimization of the transformation. Our experiment also proves that the median loss promotes the convergence while others variants fail, as shown in Tab. 6. Besides, the setting of $\mathcal{T}_{\gamma_{\mathrm{m}}}(|s|_{\mathrm{m}}) = o_m = 0.5$ is from our observation, as larger γ_m values cause early over-pruning of Gaussians, while smaller values fail to narrow the transformation. Both cases reduce performance.

The impact of the mask loss. While removing the mask loss leads to a performance decrease of 1dB, it still outperforms GS-ROR trained with the mask loss, as shown in Tab. 4.

3.2. Extra benefits from the projection-based loss

Our projection loss also promotes the Gaussian outliers moving toward the surface and diminishes the artifacts, as shown in Fig. 1.

3.3. Glossy Blender dataset

We present the relighting results with the decomposition of material and geometry on the Glossy Blender dataset [6] in Fig. 2. As there is no material ground truth, we only provide the visualization of our method. More qualitative comparisons are in Fig. 3. Our method offers smooth geometry with details preserved and thus outperforms other Gaussian-based methods regarding relighting quality. Besides, we compare our method with four NeRF-based inverse rendering methods in Tab. 2, and our method is still competitive. MII and TensoIR do not design for reflective surfaces and thus perform poorly on the Glossy dataset. Although our method has a performance gap compared to NeRO and TensoSDF, our method is far more efficient than theirs with 50% memory usage and 17% training time at most. Our method also supports real-time rendering while other NeRF-based methods fail[1]. Besides, as shown in Fig. 4, the results from our methods include sharp details that other NeRF-based methods fail to reconstruct and thus are more visually realistic.

3.4. Shiny Blender dataset

We present the decomposition of material and geometry with the relighting results on the Shiny Blender dataset [8] in Fig. 5. Due to no relighting ground truth, we show the environment map for shading as the reference. Besides, we present more qualitative comparisons of normal in Fig. 6. Due to the robustness of our method, our method provides high-quality relighting results for reflective surfaces. Additionally, We present the evaluation of surface quality in terms of Chamber distance (CD) along with NVS results in Tab. 7. Our method outperforms other methods in terms of CD, while GS-ROR achieves a higher quality than ours in some cases. The main reason is that our backbone is 2DGS

[1]NeRO and TensoSDF use the Cycles Render Engine in Blender to provide relighting results, so the FPS depends on samples per pixel, which is 1024 following the NeRO setting.

while theirs is 3DGS. Although 2DGS demonstrates superior reconstruction quality, its NVS quality is lower compared to that of 3DGS.

3.5. TensoIR synthetic dataset

We present the decomposition of material and geometry on the TensoIR synthetic dataset [2] in Fig. 7. Besides, we present more qualitative comparisons of normal in Fig. 8. Our method provides a reasonable decomposition for diffuse objects and realistic relighting results. The quantitative comparison between our method and NeRF-based methods is shown in Tab. 3, and the qualitative comparison between our method and NeRF-based methods is Fig. 9. Our method is numerically and visually competitive for diffuse object relighting while maintaining efficiency.

3.6. NeILF++ dataset

We present more comparisons between our method and other Gaussian-based inverse rendering methods on the NeILF++ [9] dataset, revealing the robustness of our method for real-world objects. As shown in Fig. 10, our method provides realistic relighting results and detailed normal for diverse real objects.

References

[1] Yingwenqi Jiang, Jiadong Tu, Yuan Liu, Xifeng Gao, Xiaoxiao Long, Wenping Wang, and Yuexin Ma. GaussianShader: 3D Gaussian splatting with shading functions for reflective surfaces. In *CVPR*, pages 5322–5332, New York, NY, USA, 2024. IEEE. 1

[2] Haian Jin, Isabella Liu, Peijia Xu, Xiaoshuai Zhang, Songfang Han, Sai Bi, Xiaowei Zhou, Zexiang Xu, and Hao Su. TensoIR: Tensorial inverse rendering. In *CVPR*, pages 165–174, New York, NY, USA, 2023. IEEE. 2, 4

[3] Brian Karis. Real shading in unreal engine 4. *Proc. Physically Based Shading Theory Practice*, 2013. 1

[4] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian splatting for real-time radiance field rendering. *ACM TOG*, 42(4):139–1, 2023. 1

[5] Jia Li, Lu Wang, Lei Zhang, and Beibei Wang. TensoSDF: Roughness-aware tensorial representation for robust geometry and material reconstruction. *ACM TOG*, 43(4), 2024. 2

[6] Yuan Liu, Peng Wang, Cheng Lin, Xiaoxiao Long, Jiepeng Wang, Lingjie Liu, Taku Komura, and Wenping Wang. NeRO: Neural geometry and BRDF reconstruction of reflective objects from multiview images. *ACM TOG*, 42(4), 2023. 2, 3

[7] Stephen McAuley, Stephen Hill, Adam Martinez, Ryusuke Villemin, Matt Pettineo, Dimitar Lazarov, David Neubelt, Brian Karis, Christophe Hery, Naty Hoffman, and Hakan Zap Andersson. Physically based shading in theory and practice. In *ACM SIGGRAPH 2013 Courses*, New York, NY, USA, 2013. Association for Computing Machinery. 1

[8] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. Ref-NeRF: Structured view-dependent appearance for neural radiance fields. In *CVPR*, pages 5491–5500, New York, NY, USA, 2022. IEEE. 3

[9] Jingyang Zhang, Yao Yao, Shiwei Li, Jingbo Liu, Tian Fang, David McKinnon, Yanghai Tsin, and Long Quan. NeILF++: Inter-reflectable light fields for geometry and material estimation. In *ICCV*, pages 3601–3610, New York, NY, USA, 2023. IEEE. 4

[10] Yuanqing Zhang, Jiaming Sun, Xingyi He, Huan Fu, Rongfei Jia, and Xiaowei Zhou. Modeling indirect illumination for inverse rendering. In *CVPR*, pages 18643–18652, New York, NY, USA, 2022. IEEE. 2

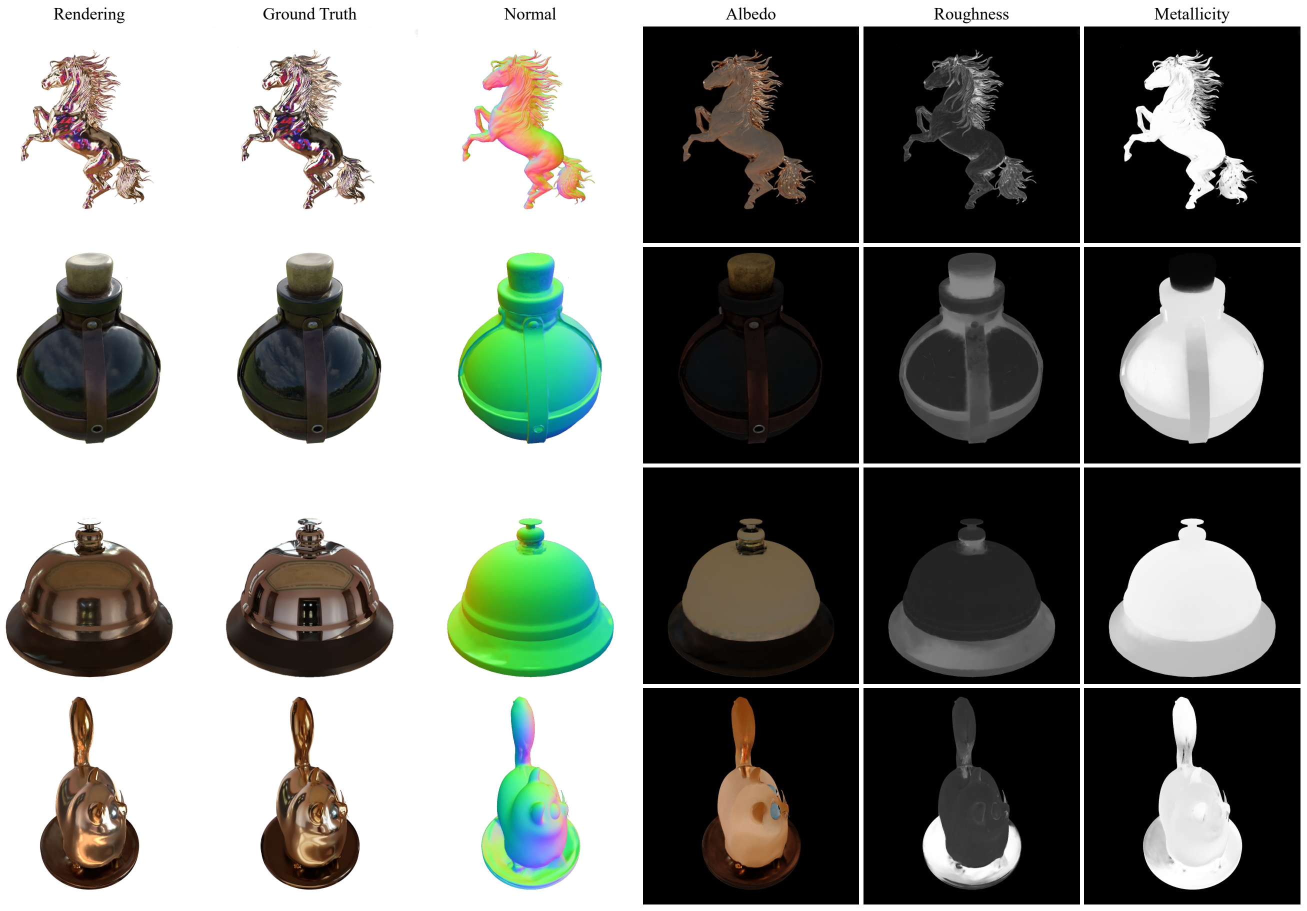

Figure 2. Decomposed maps of our method on the Glossy Blender dataset. Our method can provide a reasonable decomposition for reflective surfaces.

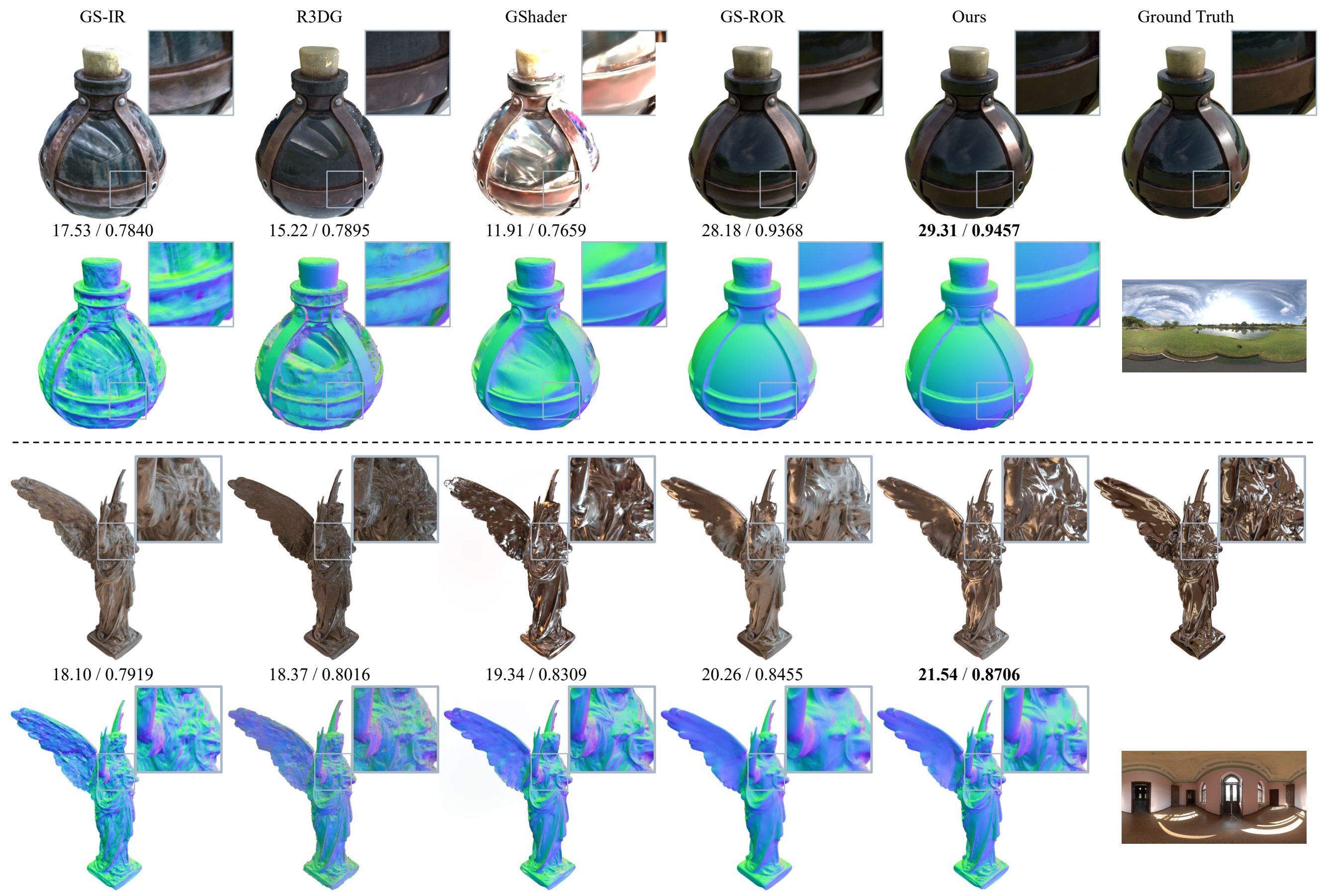

Figure 3. The qualitative comparison with Gaussian-based methods in terms of relighting results and normal on the Glossy Blender dataset. Our method provides high-quality relighting results for reflective surfaces. The PSNR/SSIM of relighting results under the current view are below the images.

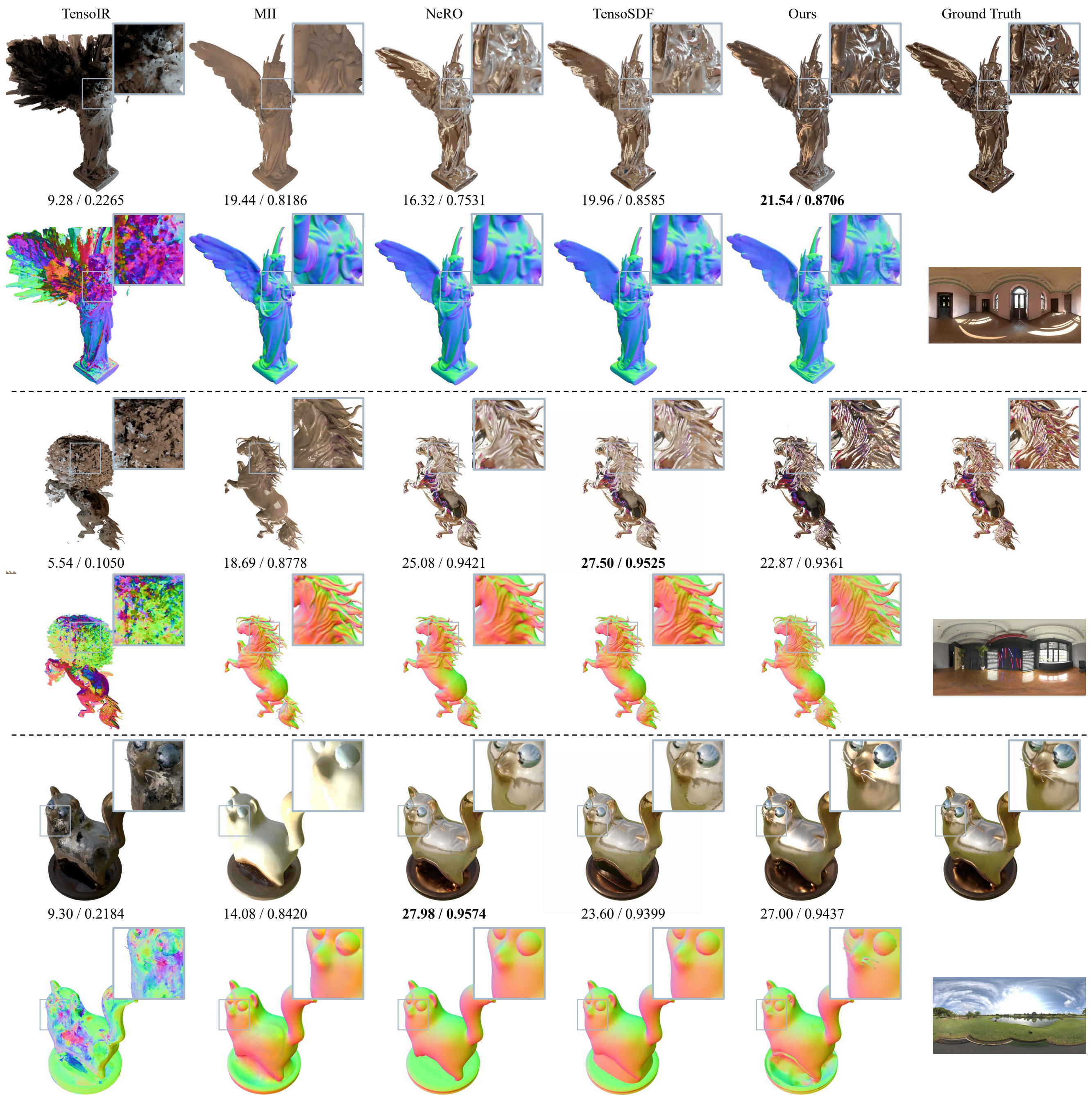

Figure 4. The qualitative comparison with NeRF-based methods in terms of relighting results and normal on the Glossy Blender dataset. Our method is competitive with these methods and outperforms in the detailed regions. The PSNR/SSIM of relighting results under the current view are below the images.

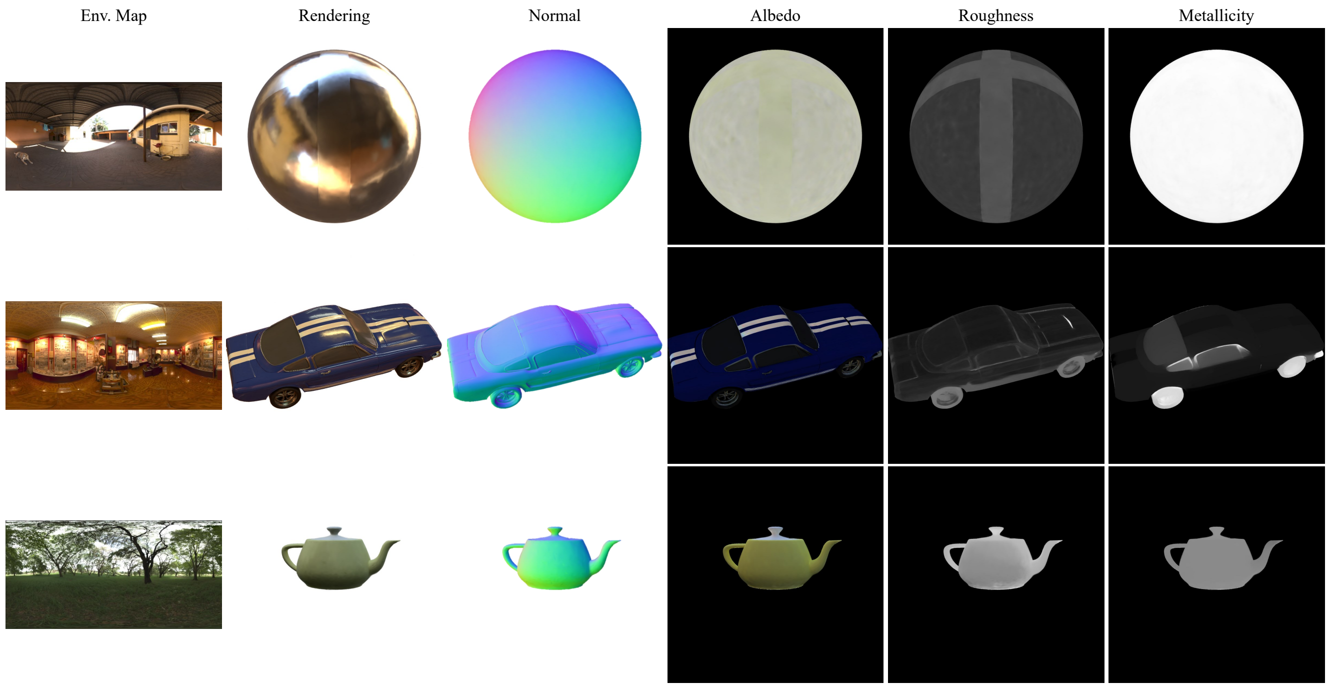

Figure 5. Decomposed maps of our method on the Shiny Blender dataset. The relighting ground truth is unavailable on the dataset, so we provide the environment map for shading as a reference.

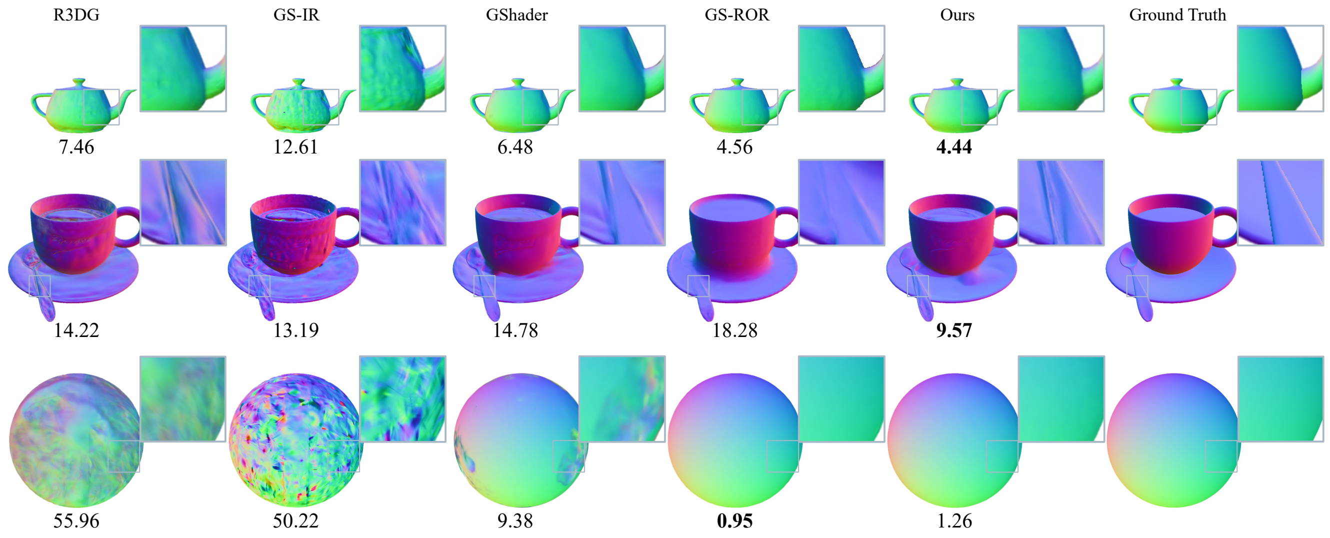

Figure 6. The qualitative comparison with Gaussian-based methods in terms of normal on the Shiny Blender dataset. The MAE under the current view is above the visualization. Our method can provide reasonable normal for diverse objects.

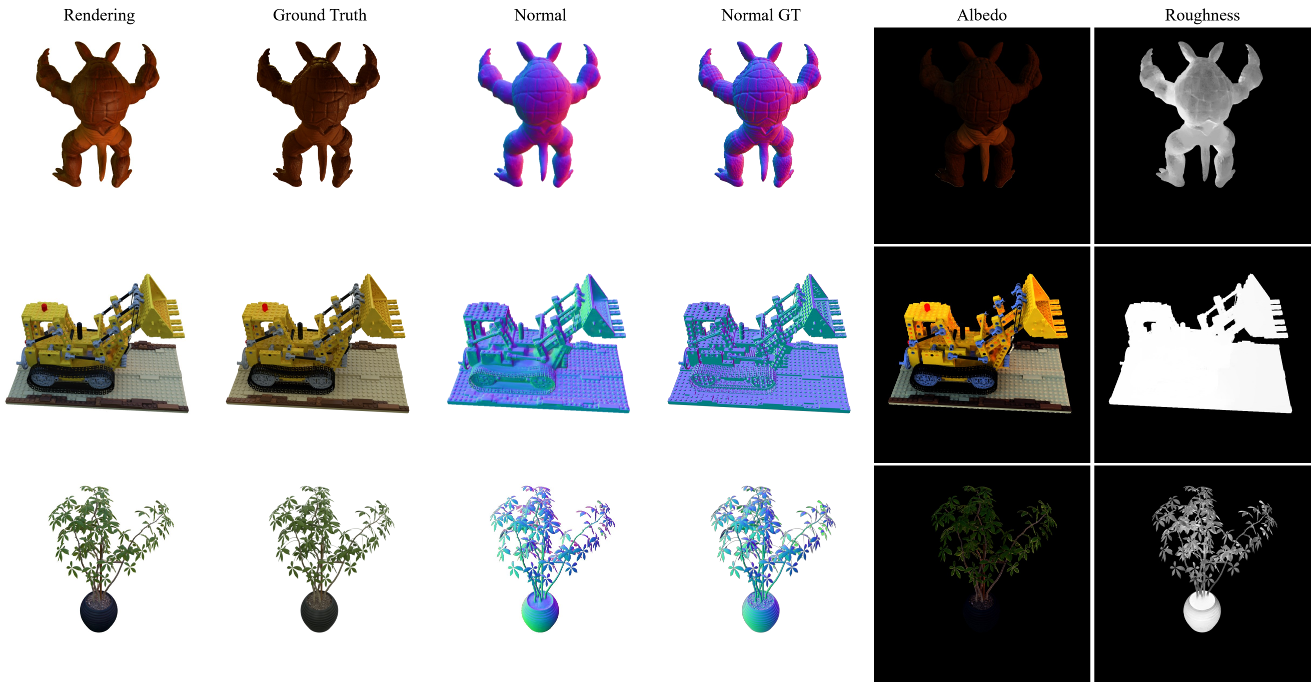

Figure 7. Decomposed maps of our method on the TensoIR synthetic dataset. Our method can provide a reasonable decomposition for diffuse objects.

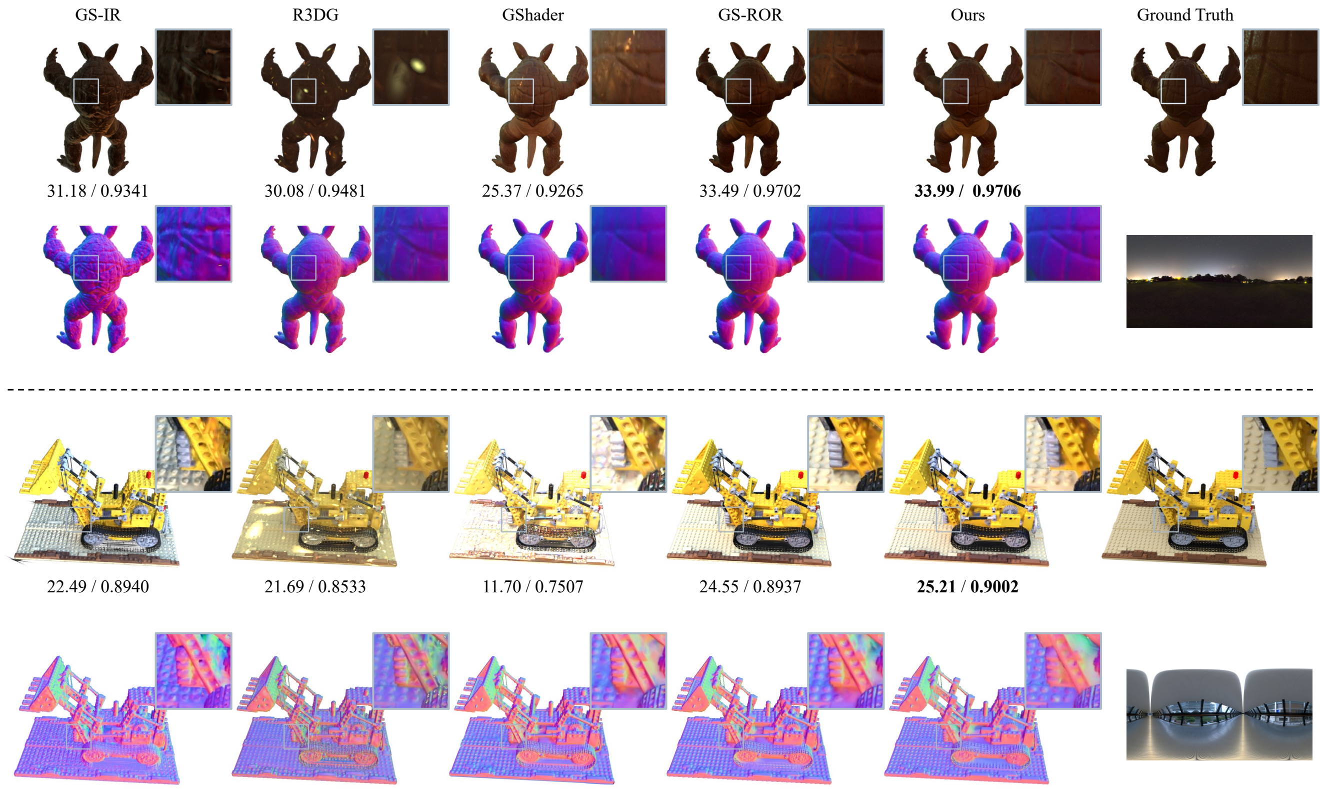

Figure 8. The qualitative comparison with Gaussian-based methods in terms of relighting results and normal on the TensoIR synthetic dataset. Our method can provide robust normal for diffuse objects. The PSNR/SSIM of relighting results under the current view are below the images.

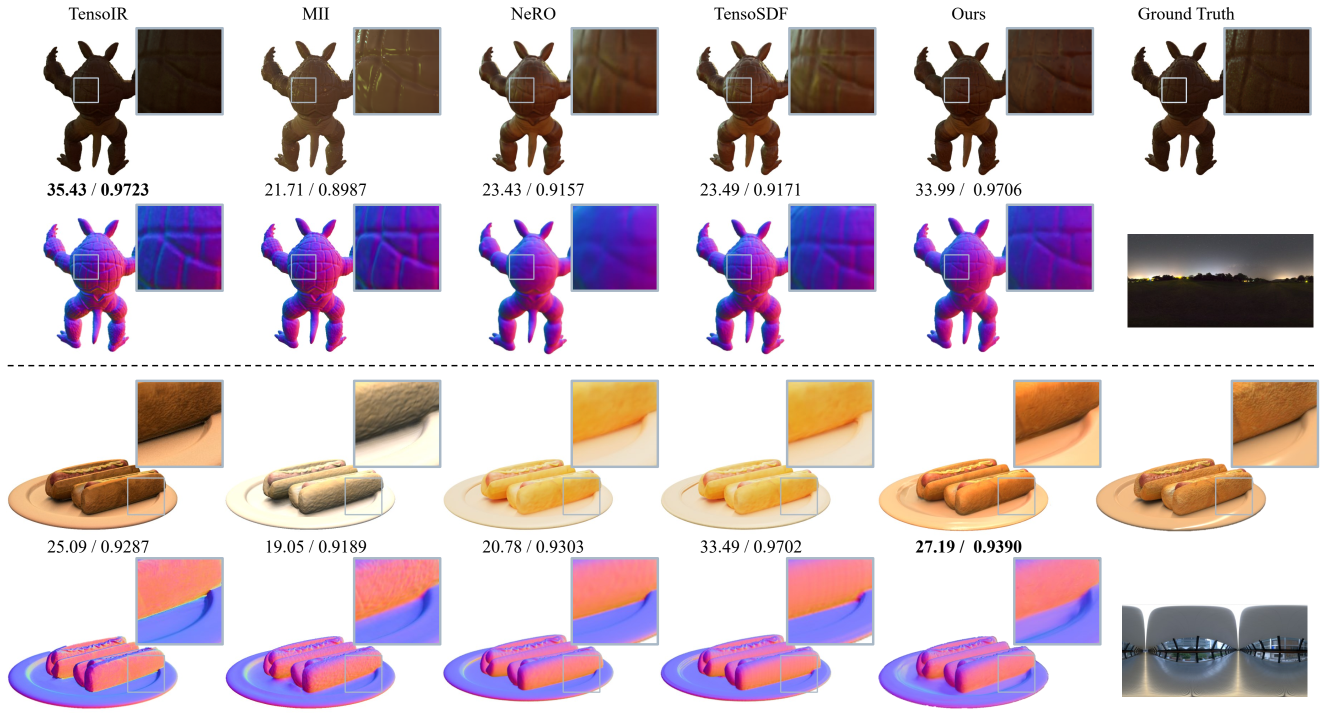

Figure 9. The qualitative comparison with NeRF-based methods in terms of relighting results and normal on the TensoIR synthetic dataset. Our method is competitive while maintaining high efficiency. The PSNR/SSIM of relighting results under the current view are below the images.

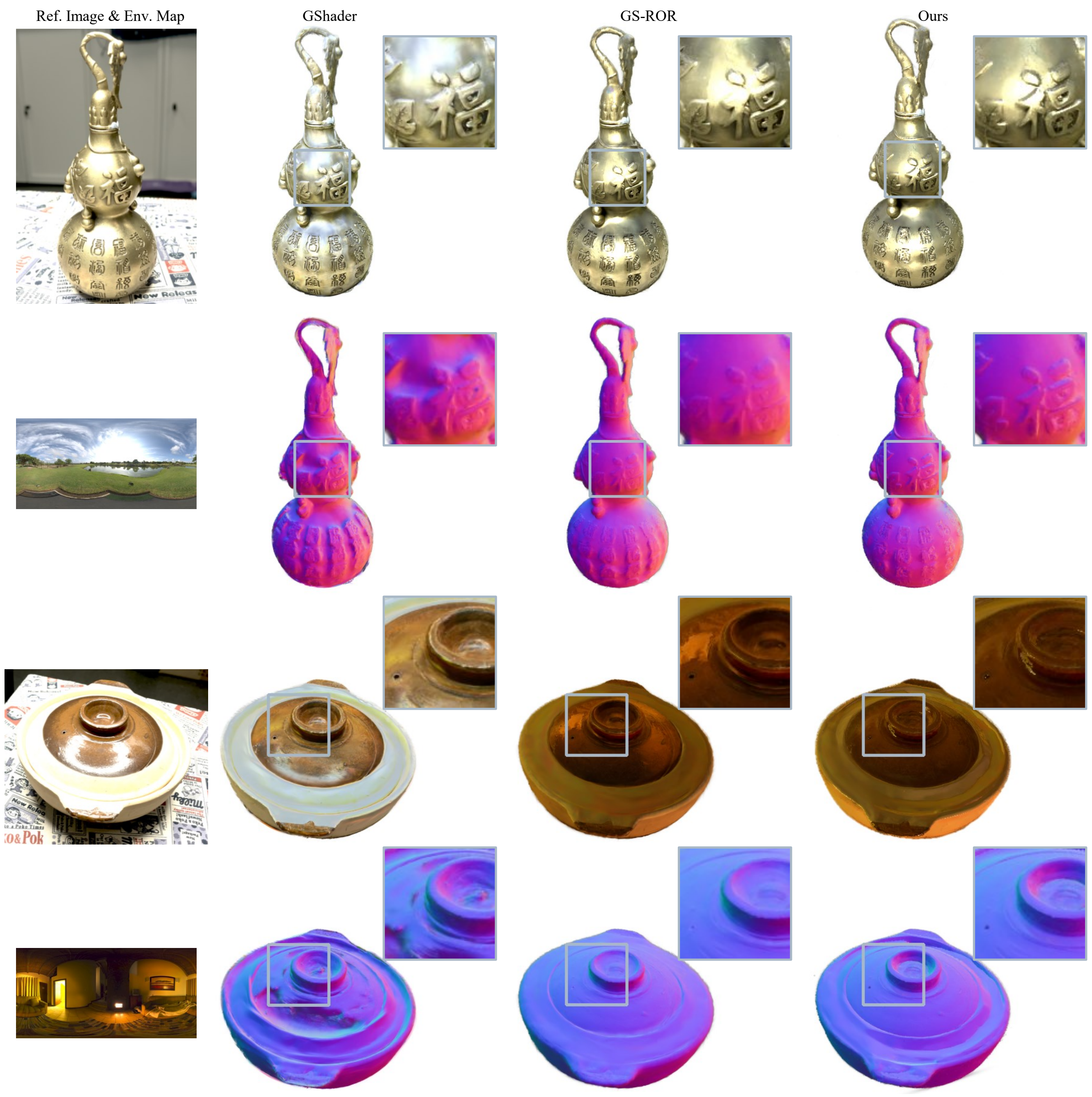

Figure 10. The qualitative comparison in terms of relighting results and normal on the NeILF++ dataset. Our method ensures the smoothness of normal while preserving details for real data, thus providing more realistic relighting results.