
PBR-SR: Mesh PBR Texture Super Resolution from 2D Image Priors (Supplementary Material)

Anonymous Author(s)

Affiliation

Address

email

1 In this supplementary material, we provide additional details and results that are not included in the
2 main paper due to space constraints. The attached video offers a brief overview of our method, along
3 with qualitative results demonstrating the performance of PBR-SR.

4 A Implementation Details

5 A.1 Dataset

6 In the main paper, we evaluate quantitative results on a collection of PBR meshes sourced from
7 [1, 2, 3]. To better evaluate PBR texture super-resolution performance, we select meshes that contain
8 rich and diverse texture in different PBR channels. Our mesh collection includes *Chinese Chandelier*,
9 *Corset*, *Damaged Helmet*, *Shoulder Strap*, *Globe*, *Under Armour Volleyball Shoe*, *Table Clock*,
10 *Medieval Chest*, *Cradle Globe*, *Armored Man*, *Lounge Chair 1*, *Lounge Chair 2*, *Old Table*, *Old Stool*,
11 *Old Chair*, and *Vintage Cozy Rocking Chair*. For evaluation, we treat the downloaded high-resolution
12 textures as ground truth. If paired low-resolution textures are provided, we use them directly as input
13 to our method. Otherwise, we generate low-resolution inputs by applying a Gaussian blur followed
14 by bicubic downsampling on the high-resolution textures. Our usage fully complies with the Royalty
15 Free License terms, as the models are only used internally for testing and are not redistributed in any
16 digital or physical form.

17 A.2 Optimization

18 The mesh is normalized to a unit size during the optimization process. The camera is positioned at a
19 distance of 3.25 units with a field of view (FoV) of 10 degrees. We use 750 views in total, sampled
20 from 15 different elevations, with 50 views evenly distributed at each elevation. For rendering
21 evaluation, we use 240 views from 6 elevations, with 40 views sampled from each elevation, ensuring
22 an even distribution across the scene. During optimization, the λ_{pix} is set to 100 and λ_{reg} is 0.5 in
23 the robust pixel-wise loss term. The weighting map is optimized in resolution of 64×64 and is
24 interpolated to \mathbf{W}_i with the same resolution as the rendering’s image when calculating \mathcal{L}_{pix} . We
25 assign different weights to the channels of the PBR texture maps in the PBR consistency loss function:
26 we set the weight to 1.0 for the diffuse $w_{\mathbf{K}^d}$, roughness $w_{\mathbf{K}^r}$, and normal map $w_{\mathbf{K}^n}$, and 0.1 for the
27 metallic map $w_{\mathbf{K}^m}$. The SSIM part is weighted by $\lambda_{\text{ssim}} = 10$ for all. The overall PBR consistency
28 loss is given a weight $\lambda_{\text{pbr}} = 10$ and the PBR TV loss is weighted by $\lambda_{\text{tv}} = 0.5$. The optimization
29 process runs for 2000 iterations. On a single NVIDIA A6000 RTX GPU, optimizing a mesh with PBR
30 textures takes around 30 minutes with 2K-to-8K resolution and less than 8 minutes with 1K-to-2K
31 resolution offline on average. The optimizable parameters are limited to the high-resolution PBR
32 textures, as we directly update them using computed gradients.

33 A.3 Baselines

34 We leverage the official implementations of SwinIR [5], HAT [4], CAMixerSR [8], StableSR [7],
 35 OSEDiff [9] and DiffBIR [6] as baselines. For N times PBR texture super resolution, we initialize
 36 SR PBR texture maps using the corresponding models trained for specific N times super resolution,
 37 if the model is available with their pretrained weights.

38 For Paint-it SR, we adopt its core deep convolutional architecture and optimization framework but
 39 introduce significant modifications to align it with the super-resolution task. Specifically, we replace
 40 its text-to-image diffusion model and Score Distillation Sampling (SDS) loss with an image super-
 41 resolution model and a pixel-wise loss function. Additionally, we modify the deep convolutional
 42 block initialization to ensure that its initial output is zero. This design enables the convolutional block
 43 to output only the residual components, which are added to the initialized SR PBR texture maps
 44 derived from interpolated LR PBR inputs. During optimization, the CNN progressively refines the
 45 residuals while maintaining fidelity to the interpolated input.

46 For CAMixerSR-FT, we make it a supervised baseline using PBR textures to fine-tune the off-the-shelf
 47 CAMixerSR weights pre-trained on natural images. We use 19 high-quality PBR meshes from the
 48 Poly Haven website [3] (no overlap with the evaluation data), each containing diffuse, ARM (ambient
 49 occlusion, roughness, metallic), and normal maps at a resolution of 4096×4096 . For training data
 50 generation, all texture maps are downsampled to 1024×1024 using bicubic interpolation. From these,
 51 we randomly extract 480×480 patches as high-resolution targets and their corresponding 120×120
 52 downsampled versions as low-resolution inputs. This process yields a total of 24,000 LR-HR texture
 53 pairs, which are used to fine-tune the $\times 4$ PBR super-resolution model. We fine-tune the pretrained
 54 CAMixerSR [8] model on each set (diffuse, ARM, or normal). The model is initialized from the
 55 official checkpoint and trained for 50K iterations using the AdamW optimizer with a learning rate
 56 of 1×10^{-4} , weight decay of 1×10^{-5} , and $(\beta_1, \beta_2) = (0.9, 0.99)$. A multi-step learning rate
 57 scheduler is used with decay milestones at 20K and 40K iterations and a decay factor of 0.5. We
 58 adopt an L1 loss in the texture (UV) space with equal weighting and no learning rate warmup. The
 59 training is conducted with exponential moving average (EMA) enabled, using a decay rate of 0.999.
 60 All experiments are performed with strict weight loading from the pretrained model, and the best
 61 checkpoint is selected based on validation performance.

62 B Additional Results

63 **Impact of Robust Pixel-wise Loss.** In the main paper, we adopt a robust pixel-wise loss to mitigate
 64 the impact of artifacts in pseudo-GT renderings, such as view inconsistency, lighting variation,
 65 and shadow misalignment. To validate the necessity of this design, we conduct an ablation by
 66 replacing our robust loss with a standard pixel-wise rendering loss: $\mathcal{L}_{\text{pix}} = \frac{1}{b} \sum_{i=1}^b \|\mathbf{I}_{\theta}^{\text{SR}} - \mathbf{I}_{\theta}^{\text{HR}}\|_2^2$,
 67 where b denotes the number of rendered views and θ represents the optimizable parameters (i.e.,
 68 the super-resolved PBR maps). This baseline assumes uniform confidence across all pixels and
 69 ignores image-specific supervision noise. As shown in Fig. X, using this naive loss often leads
 70 to overfitting in unreliable regions such as specular highlights or shadows, resulting in artifacts
 71 and inconsistent textures. In contrast, our robust formulation introduces a per-image pixel-wise
 72 weight map $\mathbf{W}_i(u, v) \in [0, 1]$ that adaptively downweights uncertain pixels. We regularize these
 73 weights toward one to ensure stable optimization: $\mathcal{L}_{\text{robust}} = \lambda_{\text{pix}} \cdot \mathcal{L}_{\text{pix}} + \lambda_{\text{reg}} \cdot \mathcal{R}(\mathbf{W})$. This design
 74 improves both the stability and quality of optimization, particularly in regions prone to multi-view
 75 inconsistencies. As shown in Table 3, our full model with robust pixel-wise loss achieves consistently
 76 higher PSNR across PBR channels and final renderings compared to the baseline using the standard
 77 pixel-wise rendering loss. These gains highlight the benefit of adaptively downweighting unreliable
 78 pixels during optimization. Fig. 5 presents a comparison of renderings under different lighting
 79 conditions from our PBR-SR method and its variant without the robust pixel-wise loss. Without
 80 the robust loss, optimization relies on a uniform pixel-wise average over multi-view supervision,
 81 which often introduces blocky artifacts. This issue is exacerbated by the PBR consistency loss, which
 82 operates in texture space using average pooling and implicitly assumes equal reliability across all
 83 pixels. In contrast, our robust loss adaptively downweights unreliable regions (e.g., shadows or
 84 view-specific lighting inconsistencies), resulting in cleaner and more physically plausible texture
 85 reconstructions.

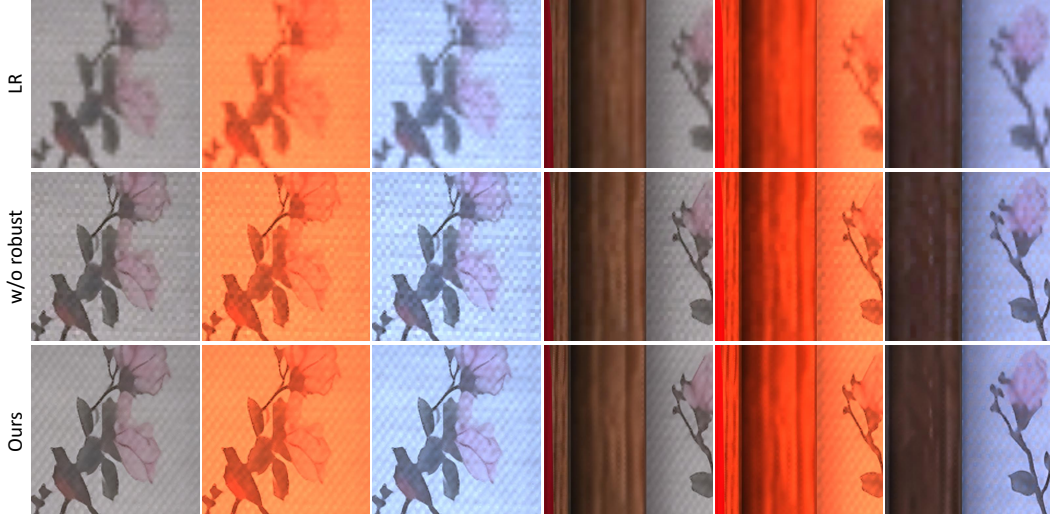


Figure 5: Comparison of renderings under different lighting from our PBR-SR and its variant without using robust pixel-wise loss. Both methods significantly improve the rendering quality from the LR PBR texture, while ours achieves high-fidelity by getting sharper and more natural details.

Table 3: Ablation on robust pixel loss in our PBR-SR pipeline. We list the quantitative comparison of PBR texture maps and renderings PSNR from $\times 4$ PBR SR results.

| Method | Albedo | Roughness | Metallic | Normal | Renderings |
|------------|--------|-----------|----------|--------|------------|
| w/o robust | 27.748 | 30.199 | 31.826 | 27.702 | 27.477 |
| Ours | 29.731 | 31.602 | 31.889 | 29.088 | 28.001 |

Table 4: Ablation on the image SR model used in our PBR-SR pipeline. We list the quantitative comparison of PBR texture maps and renderings PSNR from $\times 4$ PBR SR results.

| Method | Albedo | Roughness | Metallic | Normal | Renderings |
|-----------|--------|-----------|----------|--------|------------|
| CAMixerSR | 27.793 | 29.927 | 31.784 | 27.294 | 27.466 |
| StableSR | 27.678 | 31.025 | 32.258 | 27.995 | 26.415 |
| DiffBIR | 29.731 | 31.602 | 31.889 | 29.088 | 28.001 |

Table 5: Ablation on the rendering resolution of pseudo-GT used in our PBR-SR pipeline. We list the quantitative comparison of PBR texture maps and renderings PSNR from $\times 4$ PBR SR results. Results of $4\times$ SR on the *Table Clock* mesh are reported.

| Resolution | Albedo | Roughness | Metallic | Normal | Renderings |
|------------|--------|-----------|----------|--------|------------|
| 128 | 31.913 | 30.814 | 33.375 | 34.284 | 22.485 |
| 256 | 32.549 | 30.737 | 34.013 | 34.181 | 23.841 |
| 512 | 33.476 | 30.646 | 34.085 | 34.900 | 24.659 |
| 768 | 34.318 | 30.857 | 35.803 | 35.229 | 24.555 |
| 1024 | 36.077 | 30.644 | 36.296 | 35.620 | 24.856 |

86 **Ablation on Image SR Models in PBR-SR.** In Table 4, we compare three models used in our
87 pipeline for both initialization and rendering super resolution, and report the final optimized SR
88 results of PBR maps and corresponding renderings. DiffBIR achieves the best performance across
89 most PBR channels, including Albedo, Roughness, and Normal, as well as in the final rendering
90 PSNR. Specifically, it outperforms CAMixerSR and StableSR by a notable margin in both texture
91 fidelity and rendered appearance, indicating its superior ability to preserve structural details and

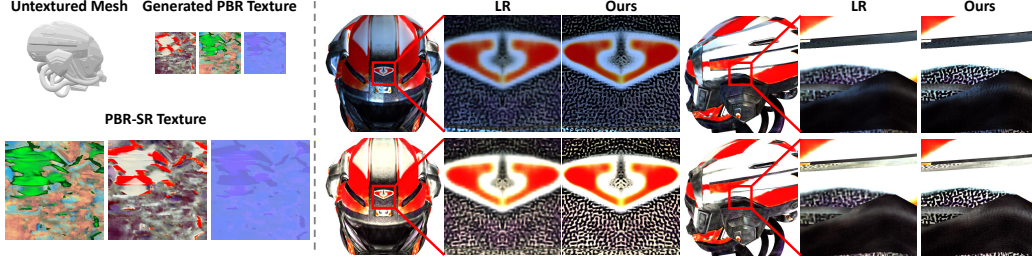


Figure 6: Texture SR for generated PBR texture. Left: we adopt Paint-it [10] to generate LR texture and use PBR-SR for $\times 4$ SR. Right: we compare renderings from LR and Ours from two viewpoints. The two rows are under different lighting conditions.

material realism in the super-resolved outputs. While StableSR performs well on the Metallic channel, it underperforms on others and results in lower rendering quality overall. Moreover, DiffBIR is computationally more efficient than StableSR, offering faster inference while maintaining high-quality outputs. This makes it a favorable choice for integration into our iterative optimization pipeline, where both accuracy and runtime efficiency are critical.

Ablation on Rendering Resolution in Optimization. We assess the impact of rendering resolution on optimization performance using the *Table Clock* model at 128×128 , 256×256 , 512×512 , 768×768 , and 1024×1024 . As shown in Table 5, increasing the resolution of the pseudo-ground truth renderings consistently improves performance across most channels. Notably, the Albedo and Metallic maps benefit the most as the resolution increases. Rendering quality also improves steadily, with PSNR rising from 22.49 to 24.86, indicating that higher-resolution renderings provide more accurate supervision signals during optimization. The Normal map shows consistent gains, reflecting enhanced geometric fidelity, while the Roughness channel exhibits only minor variation, suggesting lower sensitivity to resolution changes. Given the diminishing returns beyond 1024×1024 and the significantly increased computational and memory costs at higher resolutions, we adopt 1024×1024 as the default resolution for DiffBIR outputs in our main experiments.

Texture SR for Generated PBR Texture. We adopt Paint-it [10] to generate 1024×1024 PBR textures from an untextured mesh (*Damaged Helmet*), and apply PBR-SR to perform $4\times$ super-resolution on the generated PBR maps. As shown in Fig. 6 (right), our method successfully introduces high-frequency details that are absent in the low-resolution renderings. However, the performance of PBR-SR on generated textures is inherently limited by the quality of the input. Since the generated textures often lack fine-grained structural cues, it becomes challenging for the SR model to hallucinate or infer additional detail. Moreover, current PBR texture generation methods struggle to produce material properties that are physically consistent and visually comparable to those crafted by professional artists. We believe that future advances in generative PBR modeling could complement PBR-SR effectively. A more realistic and physically plausible generation of initial textures would allow PBR-SR to further enhance detail quality, potentially approaching the fidelity of artist-created PBR assets.

Qualitative Results on Composed Scenes. To evaluate PBR-SR on scenes with multiple objects, we composed eight meshes from the CGTrader dataset. Figure 7 compares scene renderings using the input low-resolution (LR) textures and our PBR-SR textures. Our method significantly enhances the details of each object in the scene. Additionally, we compare scene renderings under different environment lighting conditions in Figure 8. We tested three environment maps from Poly Haven [3]: *Billiard Hall*, *De Balie*, and *Beach Parking*. The results demonstrate that PBR-SR consistently improves rendering quality across all lighting scenarios.

Video. We include a supplementary video showcasing various aspects of our method. The video provides an overview of the approach, a visualization comparing the LR PBR texture maps with the PBR-SR outputs, and results under different relighting conditions. Additionally, it features comparisons with baseline methods and renderings of a composed scene, highlighting the differences between LR textures and PBR-SR textures.

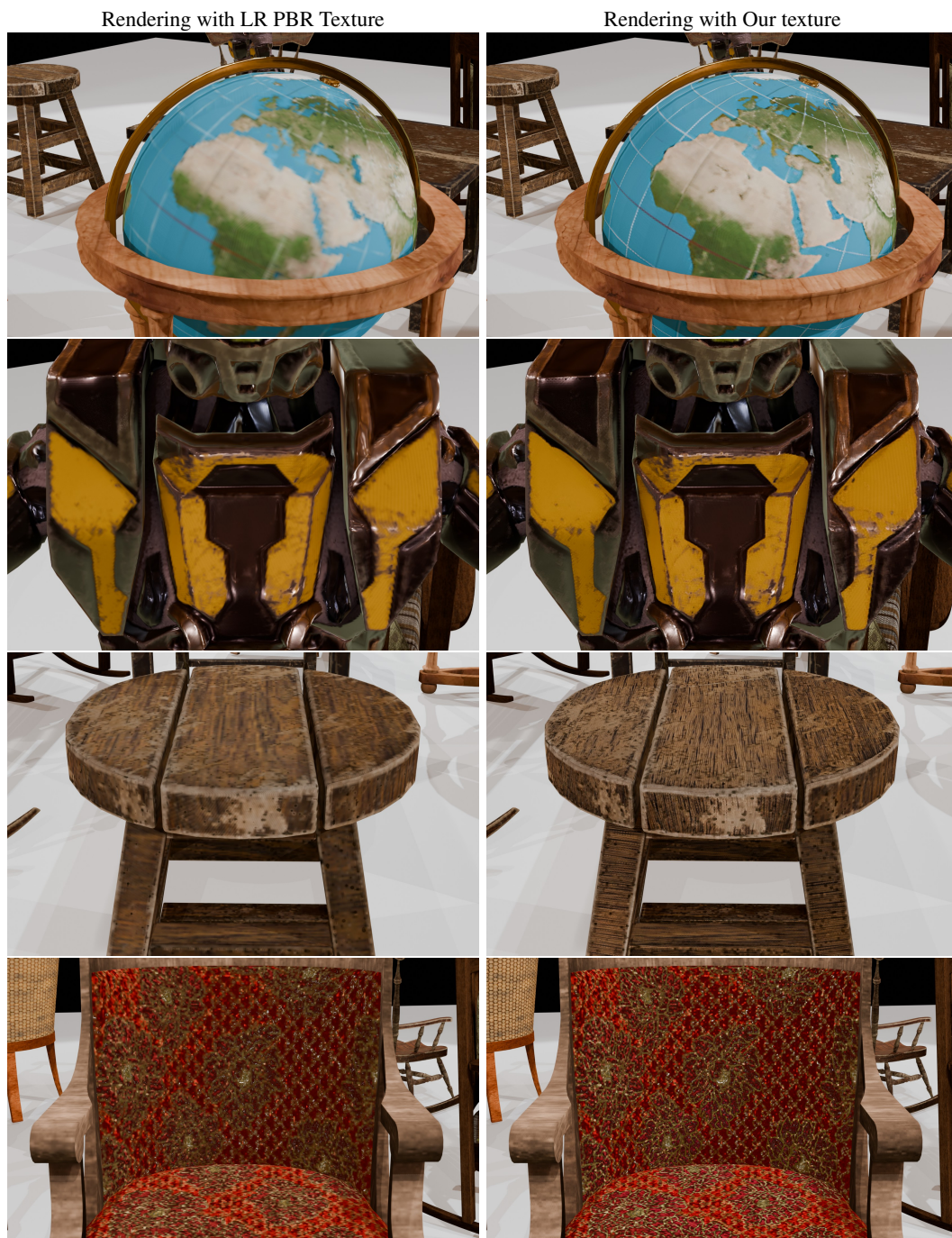


Figure 7: Comparison of scene renderings from LR textures and PBR-SR textures.

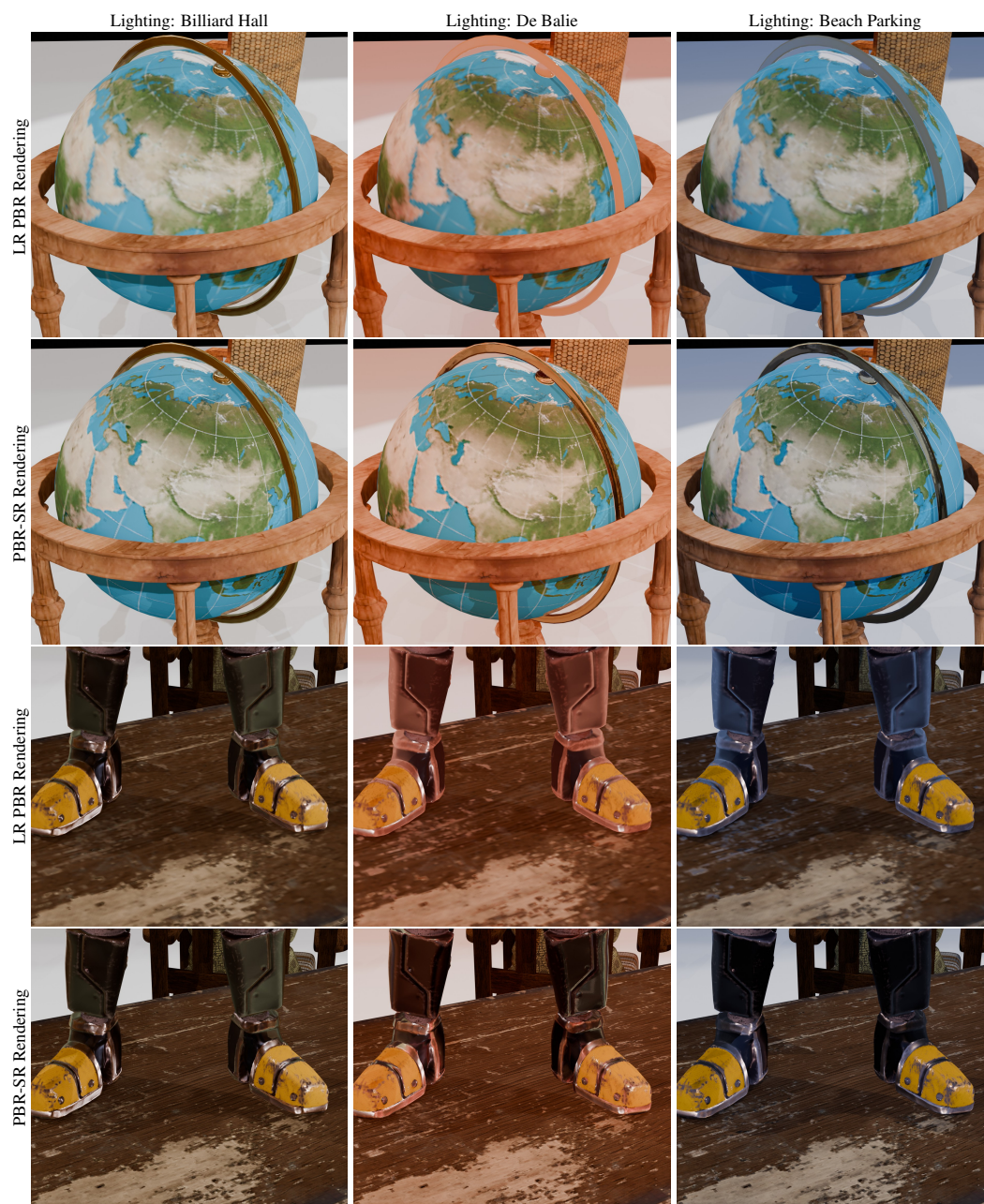


Figure 8: Comparison of scene renderings from LR textures and PBR-SR textures under novel environment lighting.

References

- [1] Cgtrader models. <https://www.cgtrader.com>.
- [2] The gltf v2.0 sample models. <https://github.com/KhronosGroup/gltf-Sample-Assets>.
- [3] Poly haven models. <https://polyhaven.com/models>.
- [4] Xiangyu Chen, Xintao Wang, Wenlong Zhang, Xiangtao Kong, Yu Qiao, Jiantao Zhou, and Chao Dong. Hat: Hybrid attention transformer for image restoration. *arXiv preprint arXiv:2309.05239*, 2023.
- [5] Jingyun Liang, Jiezhong Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1833–1844, 2021.
- [6] Xinqi Lin, Jingwen He, Ziyang Chen, Zhaoyang Lyu, Bo Dai, Fanghua Yu, Yu Qiao, Wanli Ouyang, and Chao Dong. Diffbir: Toward blind image restoration with generative diffusion prior. In *European Conference on Computer Vision*, pages 430–448. Springer, 2024.
- [7] Jianyi Wang, Zongsheng Yue, Shangchen Zhou, Kelvin C.K. Chan, and Chen Change Loy. Exploiting diffusion prior for real-world image super-resolution. 2024.
- [8] Yan Wang, Yi Liu, Shijie Zhao, Junlin Li, and Li Zhang. Camixersr: Only details need more "attention". In *CVPR*, pages 25837–25846, June 2024.
- [9] Rongyuan Wu, Lingchen Sun, Zhiyuan Ma, and Lei Zhang. One-step effective diffusion network for real-world image super-resolution. *arXiv preprint arXiv:2406.08177*, 2024.
- [10] Kim Youwang, Tae-Hyun Oh, and Gerard Pons-Moll. Paint-it: Text-to-texture synthesis via deep convolutional texture map optimization and physically-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4347–4356, 2024.