GauUpdate: New Object Insertion in 3D Gaussian Fields with Consistent Global Illumination

Supplementary Material

6. Experimental Details

We first train the Gaussian field mentioned in Sec. 3.2 for 30k steps to reconstruct the source and target scenes. Subsequently, the material hash grid is trained for 10k steps to estimate material properties and environmental illumination, utilizing the Adam optimizer [21] for optimization. During this stage, all the properties of both Gaussian fields are fixed to ensure stable decomposition. The learning rate is set to 0.001 and remains constant throughout the training, while the probability of random swapping is fixed at 0.2. The hash grid is set with 8 levels, a hashmap size of 2^{15}, and a scaling factor of 2.0 between levels, balancing multi-scale representation and granularity of the encoded data. The training process for the material hash grid takes approximately 45 mins, depending on the number of Gaussians and the scene complexity. All the experiments are conducted on a single NVIDIA RTX 4090 GPU.

Ground Truth Acquisition. To collect the ground-truth of GauUpdate-Synthetic, we record the camera poses of the rendered target scene images and use them to re-render a set of ground-truth images with the updated objects. The ground truth for real-world scenes is generated by first performing multiview reconstruction on the target scene composed of new objects with target illumination using Gaussians. Subsequently, the reconstructed Gaussians are aligned with the target field as described in Sec. 3.3 to render corresponding views for evaluation. The previews of GauUpdate-Synthetic and GauUpdate-Real datasets are shown in Fig. 7.

Figure 7. Examples of GauUpdate-Synthetic (top 2 rows) and GauUpdate-Real (bottom). Images are cropped for visualization.

7. Segment Any GAussian

Segment Any GAussian (SAGA) is an efficient Gaussian segmentation framework based on SAM [22]. It augments the pretrained 3D Gaussians with Gaussian affinity features for distilling and storing the segmentation capability of SAM. These Gaussian affinity features can be applied to indicate the probability that the Gaussians belong to the same segmentation by measuring their cosine similarity. To achieve this, SAGA employs a soft scale gate mechanism with contrastive learning to solve the problem of multi-granularity segmentation. It adopts a single layer MLP to project the features into different granularity subspaces and train these features using the correspondence distillation loss. During inference, SAGA takes as input the pixel prompt and outputs the segmented Gaussians set by evaluating the feature similarities with the query feature derived from the pixel prompt.

Our system supports two methods to prompt SAGA for segmenting the updating object, depth comparison and manually provided prompts. The illustration of depth comparison for prompting SAGA is shown in Fig. 8. The key idea of using depth comparison to guide SAGA is to identify the changing regions between the source and target fields. Since the appearance differences between the two fields can be substantial, rendering-based comparisons fail to effectively detect these variations. Therefore, we opt to compare their depth maps instead. Specifically, we first render the depth maps of the aligned source and target fields from a given viewpoint and compute a difference map by subtracting the source field's rendered depth from that of the target field. To mitigate the impact of depth misalignment of two fields caused by lighting, training viewpoints, etc., we apply a threshold to the difference map, marking only pixels with a difference exceeding the threshold as 1. Subsequently, a morphological erosion operation is performed on the difference map to eliminate noise in the background to improve robustness. Finally, we randomly sample points within the detected region to serve as prompts. If the depth differences between the source and target scenes are too noisy for effective depth comparison, we can manually specify prompts to guide SAGA in segmenting the objects that require updating. Thanks to SAGA's robust segmentation capabilities, even with manually provided prompts, fewer than three prompts are typically sufficient to achieve high-quality segmentation. The visualization of segmentation results is illustrated in the bottom row of Fig. 9.

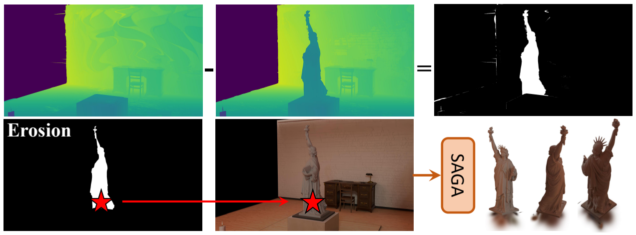

Figure 8. Illustration of depth comparison for prompting SAGA.

8. Shadow Mapping

Shadow mapping precomputes the visibility from the light source to identify whether a pixel in the camera view is in shadow. In our implementation, we treat the light source as a point light located at the brightest spot. Therefore, we begin by estimating the strongest light direction from the predicted target illumination L_i^{target}. We first convert the cubemaps of environmental light into a latitude-longitude-based equirectangular map and determine the light source as the average of the top n% brightest pixels' coordinates. We then map its latitude and longitude back to spherical coordinates and position the light source at a distance that fully encompasses the entire scene, with its orientation directed toward the center of the added object. This setup determines a light source position capable of producing reasonable shadows. Next, we follow the shadow mapping to compare the depth at the light source with the depth from the observation perspective to apply shadowing. Specifically, a depth map is first rendered from the light's perspective, recording its distances to the nearest surfaces of the scene, which forms the shadow map. During the main rendering pass, each pixel in the image plane is transformed into world space and projected into the light's view to compare its depth with the corresponding value in the shadow map, determining whether it is in shadow.

We conduct ablation studies comparing the visual effects with and without shadow mapping shown in Fig. 10. Disabling shadow mapping fails to render plausible shadows for newly added objects and the target scene due to problems like ill-posedness in PBR. This causes objects to appear as if they are floating above the surface. In contrast, enabling shadow mapping produces more visually coherent and seamless renderings.

9. Physically-Based Rendering

We follow the Physically-Based Rendering (PBR) to model how light interacts with Gaussian fields to produce realistic colors. The outgoing radiance L_o is formulated as follows given the surface point $\mathbf{p}$ and its normal $\mathbf{n}$:

$$L_o(\mathbf{p},\omega_o) = \int_\Omega f_r(\omega_i,\omega_o)\, L_i(\mathbf{p},\omega_i)\,(\omega_i\cdot\mathbf{n})\, d\omega_i\ , \quad (7)$$

where ω_i is the incoming light direction on the upper hemisphere Ω and ω_o is the outgoing lighting direction, which is opposite to the viewpoint direction. L_i is the incoming radiance and f_r is the BRDF function. We leverage the Cook-Torrance micro-facet model [9] to formulate the BRDF function, which consists of the diffuse and specular terms,

$$f_r(\omega_i,\omega_o) = \underbrace{(1-m)\frac{\mathbf{a}}{\pi}}_{\text{diffuse}} + \underbrace{\frac{DFG}{4(\omega_i\cdot\mathbf{n})(\omega_o\cdot\mathbf{n})}}_{\text{specular}}\ , \quad (8)$$

where $\mathbf{a} \in [0,1]^3$ is the albedo color, $m \in [0,1]$ is the metalness. Functions D, F, and G represent the micro-facet distribution function, Fresnel reflection, and geometry term, respectively. These terms are all determined by the albedo $\mathbf{a}$, metalness m, and roughness $\rho \in [0,1]$.

The detailed definitions of D, F, and G are as follows,

$$\begin{aligned} D(\mathbf{n},\mathbf{h}) &= \frac{\rho^4}{\pi\left(\mathbf{n}\cdot\mathbf{h}\,(\rho^4-1)+1\right)^2} \\ F(\omega_i,\mathbf{n}) &= F_0 + (1-F_0)(1-\mathbf{n}\cdot\omega_i)^5 \\ G(\omega_o,\omega_i,\mathbf{h}) &= G_1(\omega_o,\mathbf{h})\,G_1(\omega_i,\mathbf{h})\ , \end{aligned} \quad (9)$$

where $\mathbf{h}$ is the halfway vector, and G_1 and F_0 are defined as,

$$\begin{aligned} G_1(\mathbf{n},\mathbf{h}) &= \frac{1}{1+\mathbf{n}\cdot\mathbf{h}\sqrt{\rho^4+\mathbf{n}\cdot\mathbf{h}-\rho^4*\mathbf{n}\cdot\mathbf{h}}}, \\ F_0 &= (1-\mathbf{m})*0.04+\mathbf{m}*\mathbf{a} \end{aligned} \quad (10)$$

The rendering of Eq. (7) can be separated into two terms if substitute the BRDF term f_r with Eq. (8), corresponding to the Eq.(3) and Eq.(4) in Sec. 3.5.

$$\begin{aligned} L_o(\mathbf{p},\omega_o) = &\underbrace{\int_\Omega (1-m)\frac{\mathbf{a}}{\pi} L_i(\mathbf{p},\omega_i)(\omega_i\cdot\mathbf{n})\,d\omega_i}_{L_{\text{diff}}} \\ &+ \underbrace{\int_\Omega \frac{DFG}{4(\omega_i\cdot\mathbf{n})(\omega_o\cdot\mathbf{n})} L_i(\mathbf{p},\omega_i)(\omega_i\cdot\mathbf{n})\,d\omega_i}_{L_{\text{spec}}}\ . \end{aligned} \quad (11)$$

Indirect Illumination. To model the indirect illumination, we follow [10, 23] to store the occlusion $O(\theta,\phi)$ in Spherical Harmonics (SH), where (θ,ϕ) is the given view direction. The occlusion $O(\cdot)$ determines the weighting between the direct and indirect illumination of the diffuse term in Eq. (8).

Specifically, for each $\mathbf{v}_i \subset \mathcal{V}$, where $\mathcal{V}$ is the partitioned grids in 3D space, we first render depth maps for its six faces to construct a depth cubemap. This depth cubemap is then

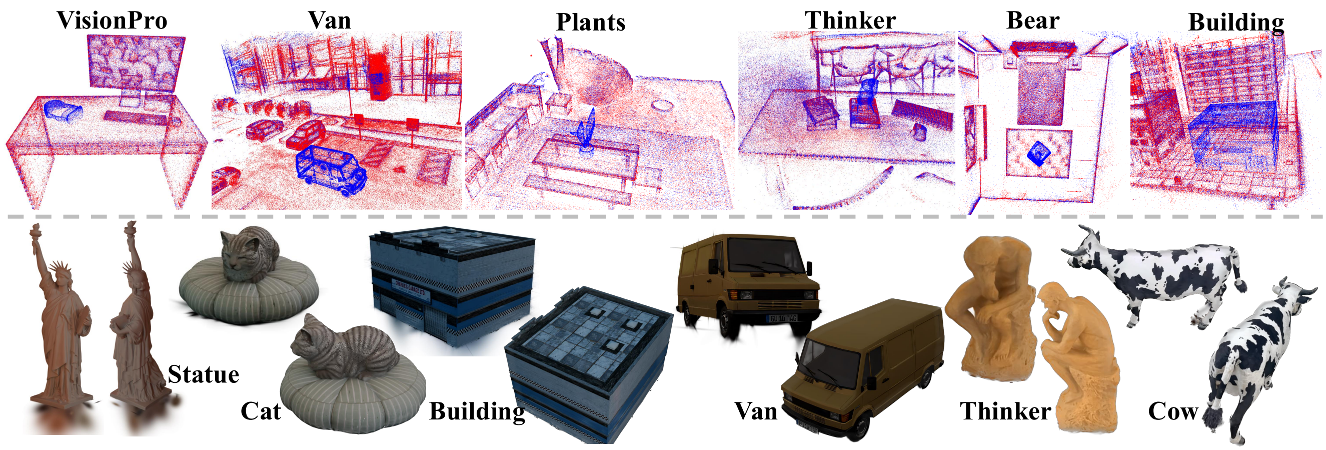

Figure 9. **Visualization of Gaussian fields registration and segmentation.** We show the results of registration in the **top** part, the source (blue) and target (red) Gaussian fields are well-aligned, with no visible misalignment at the edges. This validates the effectiveness of our registration method proposed in Sec. 3.3. The results of segmentation are shown in the **bottom**, the Gaussians of the newly inserted object are effectively segmented by SAGA.

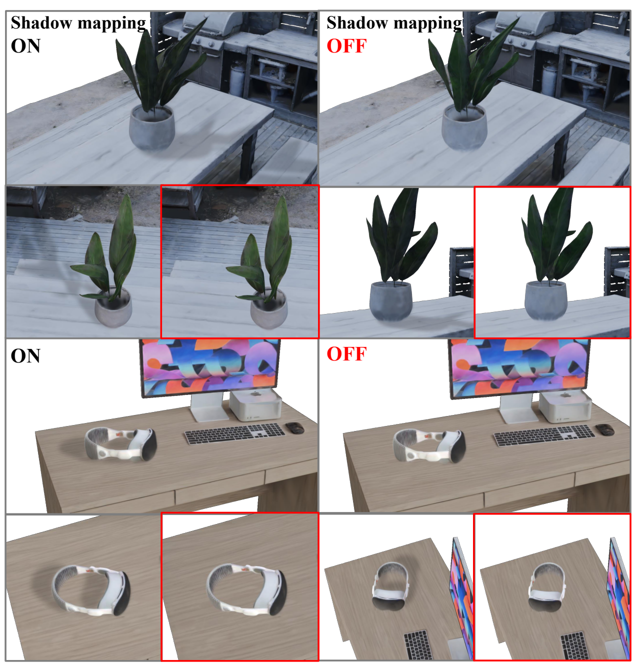

Figure 10. Ablation studies on w/ and w/o shadow mapping. Enabling shadow mapping achieves more seamless composite rendering. The results enclosed in the red box correspond to the case where shadow mapping is disabled.

binarized into an occlusion cubemap $\mathbf{O}^i = \{\mathbf{O}^i_n\}_{n=1}^6$ using a predefined depth threshold. Subsequently, we compute the SH coefficient $\mathbf{f}_i$ to represent the occlusion in SHs by convolving the occlusion cubemap $\mathbf{O}^i$ via SH basis, i.e.,

$$\mathbf{f}_{i(jk)} = \int_0^{2\pi} \int_0^{\pi} \sin\theta \, \mathbf{O}(\theta,\phi) \, Y_{jk}(\theta,\phi) d\theta d\phi, \tag{12}$$

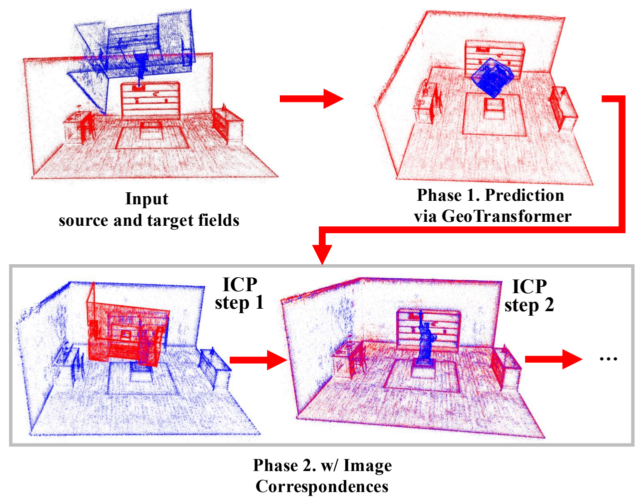

Figure 11. Illustration for the process of **Gaussian Registration**.

where $\mathbf{O}(\theta,\phi)$ is the query of occlusion cubemap from view (θ,ϕ), and $Y_{jk}(\cdot)$ is the SH basis.

Similarly, to cache the environment map for modeling indirect illumination, we also adopt SHs to store the environment cubemaps. These processes are efficiently pre-computed before the PBR, enabling real-time querying of any point within the field from the cubemaps by masked-trilinear interpolation.

10. Analysis on the Gaussian Field Registration

As described in Sec. 2.3, our registration process consists of a coarse phase to adopt the off-the-shelf point cloud registration method [30] to predict an initial transformation, and a fine phase to refine it by mapping image correspondences into 3D space. Our experiments suggest that the first phase produces a coarse transformation that may be inaccurate or even incorrect. However, this is sufficient for roughly estimating the views in the source and target fields with the highest overlap, measured by the cosine similarity between the two views. Then in the next phase, we iteratively refine the transformation in an ICP manner by each step estimating the transformation from depth-projected image correspondences using RANSAC. At the end of each iteration, the most likely overlapping viewpoints are re-calculated. We then extract their image correspondences and once again mapped them to form a 3D correspondence set. We found that this iterative approach demonstrates highly robust performance. Even if the transformation in phase one occasionally fails, the correct transformation can typically be found within a few iterations (usually fewer than five). As shown in Fig. 11, the output transformation from the first phase is relatively coarse and even be incorrect. However, after the iterative refinement, the source and target fields are aligned tightly. The visualization of Gaussian field registration is illustrated in the top row of Fig. 9, where the source (blue) and target (red) Gaussian fields are well-aligned, with no visible misalignment at the edges.

11. More Experimental Results

We showcase more results of comparison with the state-of-the-art approaches on our proposed GauUpdate-Synthetic and GauUpdate-Real in Fig. 12 and Fig. 13. We provide several cases with original-sized renderings without cropping, which offer an overall visual evaluation of seamless appearance. Our method maintains the appearance of newly inserted objects as consistent as possible with the surrounding scene, closely aligning with the ground-truth images, while other methods fail to achieve visual seamlessness.

We also provide additional qualitative comparisons in Fig. 14. The newly updated objects of our method do not exhibit albedo influenced by the lighting of the source scene, which shows superior performance over GS-IR in accurately separating intrinsic properties from environmental lighting. Consequently, our rendering closely aligns with the ground truth, achieving a significantly higher PSNR.

Figure 12. **Qualitative comparisons of Gaussian updated rendering on our proposed GauUpdate-Synthetic dataset.**

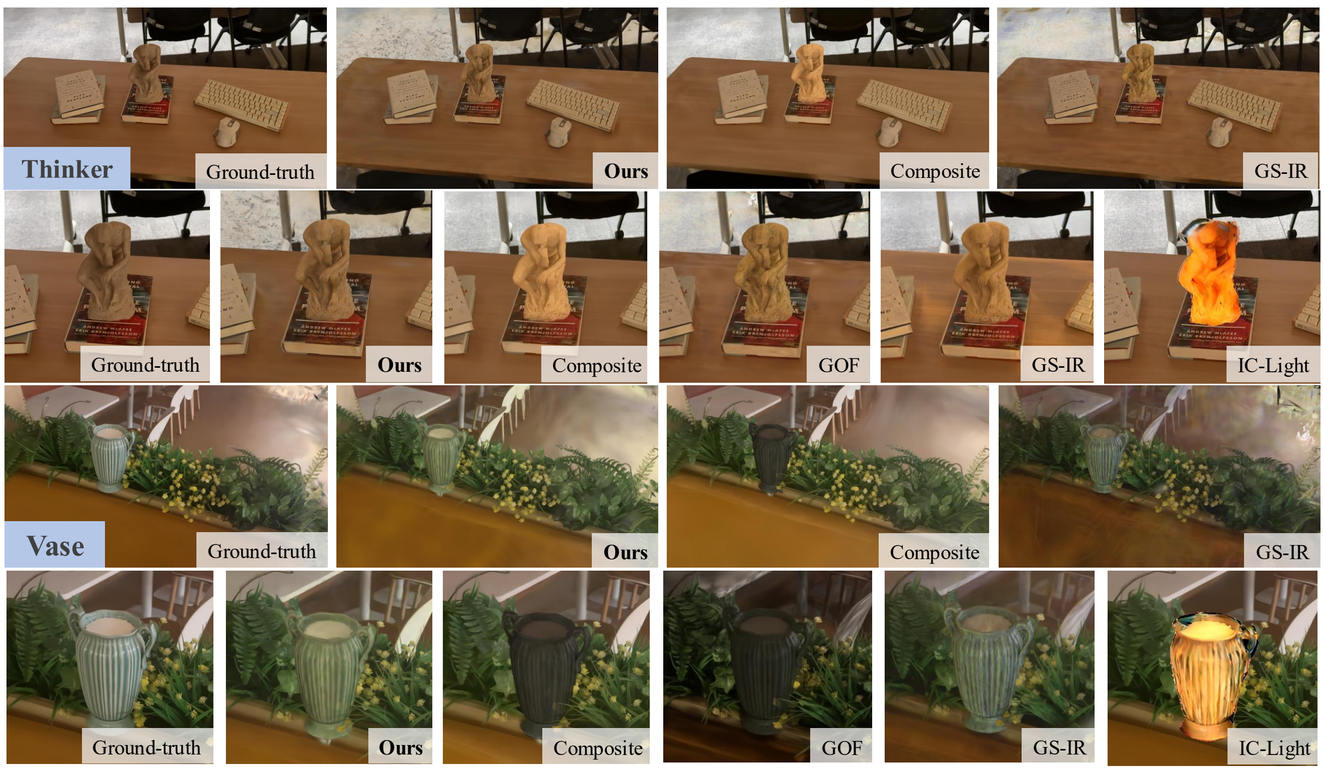

Figure 13. **Qualitative comparisons of Gaussian updated rendering on our proposed GauUpdate-Real dataset.**

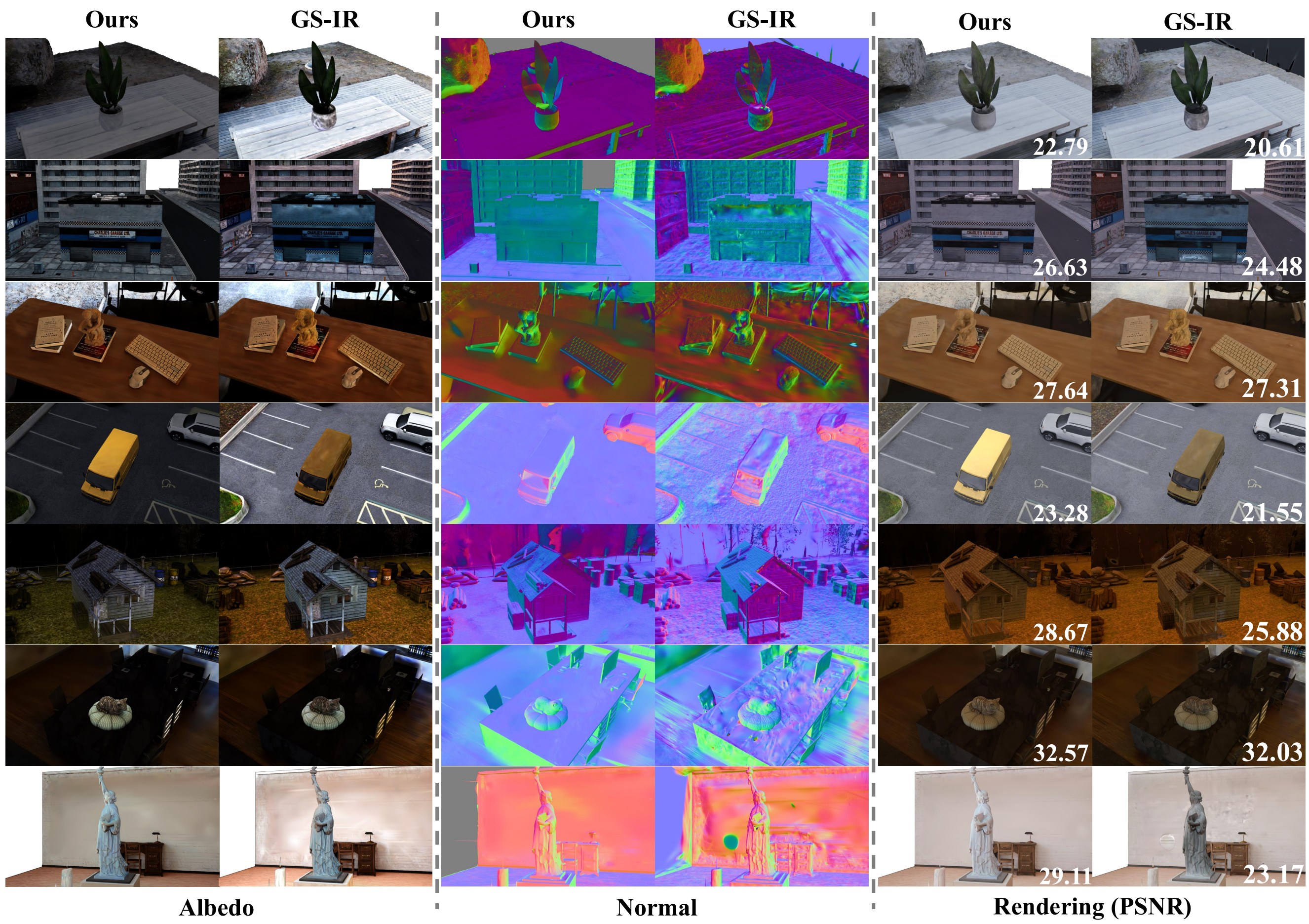

Figure 14. **We show comparisons of ours to GS-IR in terms of albedo, normal, and composite renderings of the updated Gaussian field, along with their corresponding PSNR of rendering.**