Supplemental Material
Efficient physically-based retexturing in image-space

1 Statistics of overhead compared to path tracer and Rousselle et al. [2016]

Table 1. Overhead statistics for changing the floor texture in the Living Room scene. We compare the duration of the initial rendering and re-rendering passes to a baseline path tracer. Each runs for the same number (128) of samples per pixel; timings are averaged over 5 benchmark runs.

Method	*Avg. time (s)*	*Overhead (%)*
Path Tracer	81.13	0.00%
Rousselle et al. (Initial render)	83.56	3.00%
Rousselle et al. (Re-render)	106.26	30.97%
Ours (Initial render)	92.55	14.07%

Table 2. Overhead statistics for changing the wall texture in the Living Room scene (128 spp, 5 runs).

Method	*Avg. time (s)*	*Overhead (%)*
Path Tracer	83.62	0.00%
Rousselle et al. (Initial render)	86.51	3.46%
Rousselle et al. (Re-render)	107.17	28.16%
Ours (Initial render)	93.20	11.46%

Table 3. Overhead statistics for changing both wall and floor textures in the Living Room scene (128 spp, 5 runs).

Method	*Avg. time (s)*	*Overhead (%)*
Path Tracer	81.32	0.00%
Rousselle et al. (Initial render)	84.89	4.39%
Rousselle et al. (Re-render)	106.50	30.96%
Ours (Initial render)	98.53	21.16%

Tables 1, 2, and 3 report the rendering time overhead relative to the baseline path tracer, averaged over five runs. All comparisons were performed at 128 spp. We report the overhead of the initial rendering pass for both our method and Rousselle et al. [2016], as well as the additional re-rendering pass required by Rousselle et al. [2016]. Our method incurs a moderate overhead of approximately 11–21% across all three editing tasks, primarily due to the additional AOV tracking. While the initial rendering overhead of Rousselle et al. [2016] is relatively small, their method additionally requires a costly re-rendering pass, resulting in an extra overhead of approximately 30% due to the cost for additional shading evaluations and per-pixel variance estimation. Furthermore, Rousselle et al. [2016] requires re-rendering followed by post-processing, whereas our approach supports texture edits directly after the initial render.

Table 4. Average disk space required at 1080p resolution for the Living Room scene when retexturing one object (Fig. 9 (a) and (b)).

Method	*EXR (MB)*	*JPEG (MB)*	*Scene data*
Rousselle et al. (biased)	47.55	–	Required
Rousselle et al. (unbiased)	97.5	–	Required
Ours	66.2	23.59	None

Table 5. Disk space required at 1080p resolution in Living Room scene when retexturing two objects Fig.9 (c).

Method	*EXR (MB)*	*JPEG (MB)*	*Scene data*
Rousselle et al. (biased)	46.3	–	Required
Rousselle et al. (unbiased)	94.6	–	Required
Ours	116	52.6	None

2 AOV Storage Cost

In our implementation, we use 33 AOVs that each store float16 rgb / uv data. For each scatter count h (see our eq. 9) we store the dependent and independent contributions, average UV coordinate and UV variances. Additionally, the cross terms are stored for the first three scatter counts.

Both our method and Rousselle et al. [2016] require intermediate buffers for image-space processing. Tables 4 and 5 compare their storage costs on the Living Room scene. In addition, Rousselle et al. [2016] requires scene description files for re-rendering, while our method does not. For single-object retexturing, our EXR storage cost is higher than their biased variant but lower than their unbiased variant. In the case of multiple texture edits, the storage cost of our AOVs increases.

In addition, the radiance AOVs produced by our method can be compressed using JPEG with negligible impact on the final rendered image quality, substantially reducing disk storage. AOVs containing UV information are still stored in EXR format. In contrast, the intermediate buffers used in the control variate approach of Rousselle et al. [2016] include high-frequency signed residual terms that are more sensitive to lossy compression. Image results computed using JPEG-compressed AOVs are provided in the supplemental material.

3 Post-processing cost

In Table 6, we report the runtime performance of our post-processing step at different resolutions across our test scenes. The reported total processing time includes both texture sampling and the I/O cost of reading the AOVs. Our post-processing remains efficient even at 4K resolution.

Author's Contact Information:

Table 6. Runtime performance of our post-processing at different resolutions across our test scenes.

Resolution	*Total Processing time*
640 x 640	0.719s
1080 x 1080	1.373s
2048 x 2048	3.603s
4096 x 3840	10.88s

4 Implementation

We summarize the derivation of our material segmentation, cross-term throughput and post-processing reconstruction in this section.

4.1 BSDF segmentation

We use the simplified Disney BSDF (no clearcoat and sheen, i.e., a weighted mix of microfacet lobes and diffuse / retro-reflective lobes) as an example (Fig. 2). The same principle can be extended to other BSDF models by identifying which parameters depend on the editable albedo. Our goal is to split the sampled BSDF into

$$f = f_{\text{dep}} + f_{\text{indep}},$$

where f_{dep} contains the albedo-dependent terms and f_{indep} contains the remaining terms.

4.1.1 BSDF segmentation on diffuse lobe. The diffuse lobe is purely albedo-dependent. For a Lambertian term,

$$f_{\text{diff}} = \frac{\rho}{\pi}, \qquad f_{\text{dep}} = f_{\text{diff}}, \qquad f_{\text{indep}} = 0.$$

4.1.2 BSDF segmentation of microfacet lobes. For the microfacet lobe, we split its Fresnel term into albedo-dependent and albedo-independent components. The Cook-Torrance microfacet model is given by

$$f_{\text{refl}}(\omega_o, \omega_i) = \frac{D(\omega_h)G(\omega_o, \omega_h, \omega_i)F(\omega_o, \omega_h)}{4|\omega_g \cdot \omega_o|\,|\omega_g \cdot \omega_i|}.$$

In our example material, we define metallic m, albedo ρ, index of refraction η, and specular tint t. The Fresnel term blends the dielectric Fresnel F_{diel} with the Schlick approximation,

$$F = (1-m)F_{\text{diel}} + m\left(R_0 + (\mathbf{1} - R_0)w_F\right), \tag{1}$$

where the specular reflectance parameter R_0 blends albedo and metallic is defined as $R_0 = m\rho + (1-m)F_0 t$ with reflection coefficient $F_0 = \left(\frac{\eta-1}{\eta+1}\right)^2$, and $w_F = (1 - \omega_i \cdot \omega_h)^5$. To segment the reflection lobe, we split R_0 into

$$R_{0,\text{dep}} = m\rho, \qquad R_{0,\text{indep}} = (1-m)F_0 t,$$

so that $R_0 = R_{0,\text{dep}} + R_{0,\text{indep}}$. Substituting this into Eq.1 gives

$$F = (1-m)F_{\text{diel}} + m\left(R_{0,\text{dep}} + R_{0,\text{indep}} + (\mathbf{1} - R_{0,\text{dep}} - R_{0,\text{indep}})w_F\right),$$

We segment this Fresnel term into the albedo-independent part

$$F_{\text{indep}} = (1-m)F_{\text{diel}} + m\left(R_{0,\text{indep}} + (\mathbf{1} - R_{0,\text{indep}})w_F\right),$$

and the albedo-dependent part

$$F_{\text{dep}} = m(1 - w_F)R_{0,\text{dep}}.$$

Thus the reflection lobe is segmented as

$$f_{\text{refl}}^{\text{dep}} = \frac{D(\omega_h)G(\omega_o, \omega_h, \omega_i)F_{\text{dep}}}{4|\omega_g \cdot \omega_o|\,|\omega_g \cdot \omega_i|}, \tag{2}$$

$$f_{\text{refl}}^{\text{indep}} = \frac{D(\omega_h)G(\omega_o, \omega_h, \omega_i)F_{\text{indep}}}{4|\omega_g \cdot \omega_o|\,|\omega_g \cdot \omega_i|}. \tag{3}$$

4.2 Post-processing reconstruction

In post-processing, we sample the edited texture at the stored UV statistics for each hit count. Let a_j be the new sampled texture at the j-th retextured hit. We denote the radiance AOVs accumulated with the dependent, independent, and cross-term throughputs for hit count h by $A_{\text{dep}}[h]$, $A_{\text{indep}}[h]$, and $A_{\text{cross}}[h]$. The reconstructed radiance is

$$I_h = A_{\text{dep}}[h] \prod_{j=1}^{h} a_j + A_{\text{indep}}[h] + \Delta_{\text{cross}}[h]. \tag{4}$$

The final pixel value is obtained by summing over hit counts,

$$I = A_{\text{full}}[0] + \sum_{h=1}^{H} I_h,$$

where $A_{\text{full}}[0]$ is the radiance AOV for paths with no retextured hit.

4.3 Cross-term throughput

Here we explain details on calculating the cross-term throughputs in Eq. 8c. For compactness, we show the subpath containing only retextured hits. At other vertices, Eq. 8c multiplies the same ordinary BSDF weight into T_{full}, T_{dep}, T_{indep}, and all cross-term throughputs. Let $d_j = f_{\text{dep},j}$ and $i_j = f_{\text{ind},j}$ at retextured hit count index j.

For a path with two retextured hits,

$$(d_1 + i_1)(d_2 + i_2) = \underbrace{d_1 d_2}_{\text{dep.}} + T_{\text{cross}}^{(2)}[1] + T_{\text{cross}}^{(2)}[2] + \underbrace{i_1 i_2}_{\text{indep.}},$$

where

$$T_{\text{cross}}^{(2)} = (d_1 i_2,\ i_1 d_2).$$

For a path with three retextured hits,

$$\prod_{j=1}^{3}(d_j + i_j) = \prod_{j=1}^{3} d_j + \sum_{k=1}^{6} T_{\text{cross}}^{(3)}[k] + \prod_{j=1}^{3} i_j,$$

where

$$T_{\text{cross}}^{(3)} = (d_1 d_2 i_3,\ d_1 d_3 i_2,\ d_1 i_2 i_3,\ d_2 d_3 i_1,\ d_2 i_1 i_3,\ d_3 i_1 i_2).$$

The cross-term contribution in Eq. 4 follows the same polynomial terms as above. For two retextured hits,

$$\Delta_{\text{cross}}[2] = A_{\text{cross}}^{(2)}[1]a_1 + A_{\text{cross}}^{(2)}[2]a_2.$$

For three retextured hits,

$$\Delta_{\text{cross}}[3] = A_{\text{cross}}^{(3)}[1]a_1 a_2 + A_{\text{cross}}^{(3)}[2]a_1 a_3 + A_{\text{cross}}^{(3)}[3]a_1 + A_{\text{cross}}^{(3)}[4]a_2 a_3 + A_{\text{cross}}^{(3)}[5]a_2 + A_{\text{cross}}^{(3)}[6]a_3.$$

This allows the post-processing stage to reconstruct the cross-term throughput by combinatorially multiplying the sampled textures

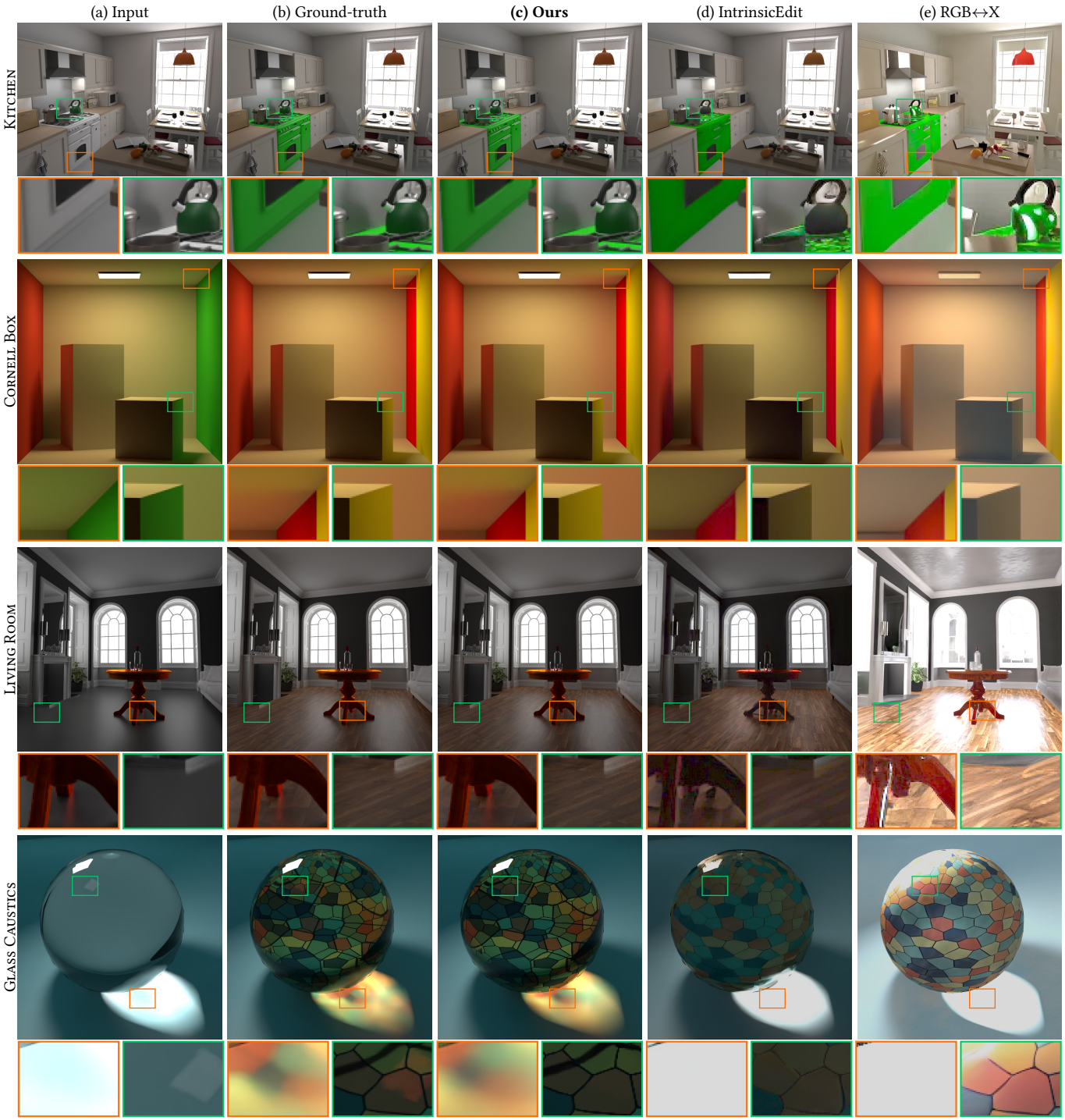

Fig. 1. We compare our results with RGB ↔ X [Zeng et al. 2024] and Intrinsic Edit [Lyu et al. 2025]. We evaluate RGB ↔ X [Zeng et al. 2024] and Intrinsic Edit [Lyu et al. 2025] using their official open-source implementations with default settings and ground-truth albedo channel. While diffusion-based methods achieve comparable results on directly seen surfaces in some cases, our approach shows improved robustness across all scenes.

with the radiance cross-term AOVs. In practice, computing the cross-term throughputs up to retextured hit count 2 already shows a good result.

5 Comparison with diffusion-based methods

Complementing the glass sphere example in Fig. 10 of the main paper, Fig. 1 provides detailed comparisons with diffusion-based

methods across a broader set of scenes. These methods introduce unpredictable changes to geometry (Kitchen), color, and luminance (all scenes), and material appearance (the glass in Glass Caustics no longer resembles glass). Furthermore, they fail to capture complex indirect illumination, such as the color bleeding in Cornell Box or the textured caustics in Glass Caustics.