TRIRENDER: SCALING NEURAL RENDERING VIA REWEIGHTED HIERARCHICAL ATTENTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Transformer-based neural renderers can synthesize globally illuminated images of unseen scenes, but dense attention becomes increasingly costly as triangle counts grow. Moreover, tessellation increases the number of triangle tokens, whose features and attention logits may remain similar to those of the original triangles. Nonuniform tessellation can therefore shift softmax attention toward subdivided regions and alter predictions even when the surface geometry is unchanged. This token-count dependence also affects hierarchical attention, where coarse tokens replace groups of triangles to reduce computational cost. We propose TriRender, a two-stage neural rendering pipeline that addresses these coupled challenges through reweighted hierarchical attention. Inspired by continuous attention, we use represented surface area as a quadrature weight, giving triangles and clusters a common measure across discretizations and hierarchy levels. This reduces the dependence of attention on token count when coarse and fine representations are combined. We couple this reweighting with a spatially cohesive 3D hierarchy, where grouping nearby triangles helps limit information loss when they are represented by a single coarse token. Regenerating coarse tokens from their constituent triangle features at each layer keeps them tied to local surface regions without maintaining independent cluster states. Using this hierarchy, the view-independent encoder combines attention to all coarse tokens with sliding-window triangle attention to encode global light transport, while the view-dependent decoder retains coarse global context and retrieves fine triangle features relevant to each image patch. Experiments demonstrate improved scalability to higher triangle counts and a favorable balance between rendering quality and inference efficiency.

1 INTRODUCTION

Traditional physically based rendering synthesizes images with global illumination by simulating light transport through a scene. Recently, RenderFormer Zeng et al. (2025) introduced a transformer-based neural rendering pipeline that directly predicts globally illuminated images of unseen triangle meshes without per-scene training or fine-tuning. However, its dense attention among triangles and between image patches and triangles incurs substantial computational cost, limiting its ability to scale to higher triangle counts.

Moreover, the same continuous surface can be tessellated in different ways. The resulting triangle sequences may differ in length, yet should produce similar results. Triangles produced by tessellation have feature representations similar to those of the original triangle and therefore may retain similar attention logits despite having smaller areas. When tessellation is nonuniform across a scene, softmax normalization can substantially shift the attention distribution toward more densely tessellated regions, potentially degrading rendering quality. This bias could also arise within a single scene containing objects at different mesh resolutions. These challenges motivate a novel neural rendering architecture that supports higher triangle counts with reduced sensitivity to changes in surface discretization.

We propose TriRender, a two-stage neural rendering pipeline with reweighted hierarchical attention. Inspired by continuous-attention formulations Calvello et al. (2025); Berner et al. (2026), we formulate attention for triangle and image-patch queries as a weighted integral over the scene surface. To

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

approximate this integral with discrete triangle tokens, we use the surface area represented by each token as its quadrature weight, implemented as a log-area bias in the attention logits. This provides a common surface measure for attention across different discretizations. Building on this formulation, we construct a 3D hierarchy, grouping triangles into spatially cohesive clusters and representing each cluster with a single coarse token. Each coarse token uses its cluster's total area as the quadrature weight, extending the same weighting rule across levels. Within-cluster attention logit variation directly can introduce error in coarse approximation Zhou et al. (2026a), motivating our spatially cohesive clustering. Hierarchical attention then aggregates global information through a compact set of coarse tokens and retrieves fine triangle features where needed. The resulting pipeline supports efficient rendering at high triangle counts with reduced dependence on surface discretization.



Figure 1: TriRender scales transformer-based neural rendering to higher triangle counts through reweighted hierarchical attention. The example scene shown here contains 540,127 triangles, of which 524,288 form the painting on the wall.

Specifically, within each mesh component, we order triangles along a Hilbert curve Hilbert (1891) and then group consecutive triangles into spatially cohesive clusters with comparable surface areas. Each cluster will be represented by a coarse token obtained by pooling its triangle features. Spatially cohesive grouping helps limit information loss during this aggregation. Meanwhile, constraining cluster surface areas reduces dependence on tessellation and helps avoid clusters spanning overly broad spatial regions. To mitigate spatial jumps across component boundaries, we then reorder all clusters in the scene along a Hilbert curve using their area-weighted centroids. On this reordered sequence, the view-independent encoder combines sliding-window attention among triangles with attention to all cluster tokens, allowing information to propagate across the scene. At each layer, cluster representations are regenerated from triangle features by area-weighted pooling, avoiding independently maintained cluster states. For each image patch, the view-dependent decoder selects the area-reweighted top-k clusters together with clusters containing directly visible triangles, and replaces their coarse tokens with all their constituent triangle tokens. The remaining clusters provide global context. We evaluate TriRender across a range of triangle counts, demonstrating improved scalability and a favorable balance between rendering quality and inference efficiency.

Concretely, we make the following contributions.

- We introduce surface-area reweighting for transformer-based rendering, reducing sensitivity to surface discretization and improving generalization to higher-resolution meshes.
- We construct a spatially cohesive 3D hierarchy and regenerate coarse cluster tokens at each layer by area-weighted pooling of their constituent triangle tokens. This keeps each coarse token tied to its local surface region while avoiding maintaining cluster states.
- We employ distinct forms of reweighted hierarchical attention for both stages, substantially reducing inference time while maintaining rendering quality.

2 Related Work

2.1 Neural Rendering

Neural rendering uses neural networks to approximate scene appearance and light transport (Tewari et al., 2022).. A well-known example is NeRF, which fits a continuous radiance field to multiview images of a scene for novel-view synthesis (Mildenhall et al., 2020). Other scene-specific representations learn responses to changes in lighting, materials, or geometry, supporting relighting and dynamic-scene rendering (Rainer et al., 2022; Raghavan et al., 2023; Zheng et al., 2024). Such representations capture rich appearance and lighting effects, but their dependence on scene-specific training or precomputation limits direct reuse in unseen scenes.

Neural rendering methods based on screen-space scene information learn shading from per-pixel attributes in G-buffers, enabling prediction across scenes (Nalbach et al., 2017; Xin et al., 2020). Recent methods use diffusion priors to synthesize realistic images conditioned on screen-space geometry and material buffers, together with lighting information (Zeng et al., 2024; Liang et al., 2025). However, visible-surface G-buffers omit off-screen and occluded geometry, limiting the prediction of shadows and reflections from these buffers alone (McGuire & Mara, 2014). This limitation motivates neural renderers that reason about the full scene to reproduce the global interactions captured by physically based rendering.

Learning light transport from 3D scene descriptions provides access to geometry beyond the current view. Hermosilla et al. (2019) learn point-based representations that predict indirect illumination from geometry, materials, and direct lighting, generalizing across shapes, materials, and lighting conditions. Xu et al. (2026) learn a generalizable 3D light transport embedding, primarily for diffuse indirect illumination, and demonstrate adaptation to glossy transport. In their diffuse GI settings, both methods use the predicted indirect illumination together with separately computed direct lighting to synthesize images. RenderFormer (Zeng et al., 2025) and RenderFormer-V2 (Zeng et al., 2026) jointly model direct and indirect illumination through an end-to-end mapping from scene primitives to images, without per-scene training. We build on this paradigm with attention over coarse and fine scene representations. Unlike RenderFormer-V2, which augments local attention with pooled summaries but retains dense image-to-scene attention, we let each image patch selectively retrieve fine triangles while retaining coarse context elsewhere. We further weight coarse and fine tokens by their represented surface area to account for nonuniform mesh discretization.

2.2 Efficient Sparse Attention

Efficient attention methods reduce the quadratic cost of dense self-attention by restricting token interactions or approximating them with compact representations (Tay et al., 2022). Local windows preserve neighborhood interactions at low cost, but do not directly connect distant tokens within a layer. Longformer (Beltagy et al., 2020) supplements sliding windows with a small set of globally connected tokens, providing communication across the sequence. Swin Transformer (Liu et al., 2021) instead alternates window partitions across successive blocks, allowing information to pass between neighboring windows while keeping attention computation local. These designs extend the context available to local interactions without computing all token pairs.

Hierarchical attention offers a complementary way to incorporate broader context. Focal Attention (Yang et al., 2021) combines fine interactions among nearby tokens with pooled representations covering larger regions, so that each query can use both local detail and distant context. H-Transformer-1D (Zhu & Soricut, 2021) recursively coarsens sequence representations and constructs a multilevel attention approximation inspired by hierarchical matrices, achieving linear time and memory complexity. In both cases, coarse representations reduce the number of interactions needed to cover a broad context while preserving fine-level computation locally.

Coarse representations can also guide content-adaptive selection of fine interactions. Native Sparse Attention (Yuan et al., 2025) uses compressed context to select token blocks and combines compressed, selected, and sliding-window attention through learned gates. Log-linear Sparse Attention (LLSA) (Zhou et al., 2026b) progressively selects relevant blocks across hierarchy levels and augments the selected fine keys and values with coarse representations collected during routing. This preserves information beyond the selected fine blocks. SVG-EAR (Zhou et al., 2026a) uses key/value cluster centroids to approximate skipped blocks and allocates exact computation according to estimated approximation error per unit cost. Its routing accounts for the error of coarse approximation rather than attention magnitude alone, highlighting the importance of the quality of coarse representations when selecting fine interactions.

We apply these local–global and selective-attention principles to neural rendering: the view-independent encoder combines sliding-window triangle interactions with coarse scene context, while the view-dependent decoder retrieves fine triangle features according to each image-patch query.

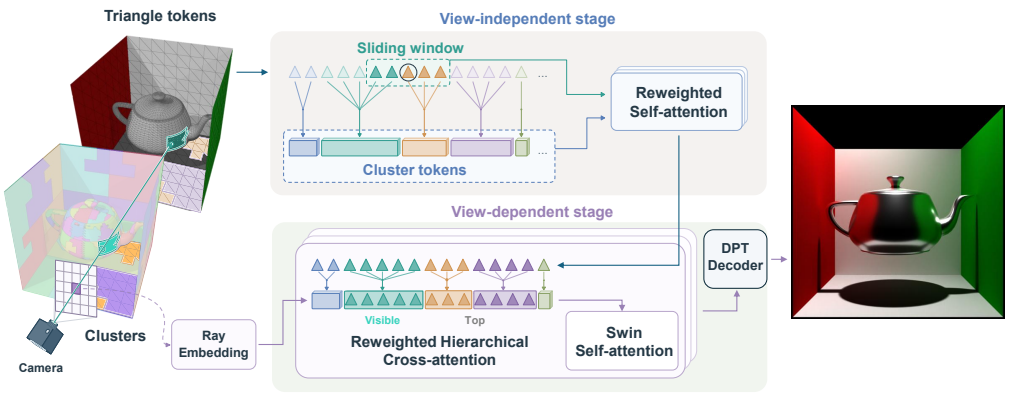

Figure 2: **Overview of our rendering pipeline.** We group non-emissive scene triangles into spatially cohesive clusters (Sec. 4.1). Stage 1 combines sliding-window attention over triangle sequence with global cluster context to encode view-independent scene features. In Stage 2, ray patch queries aggregate scene features through hierarchical cross-attention, using triangle tokens from clusters selected by visibility or top-k retrieval and coarse tokens from the remaining clusters. Both stages apply surface-area reweighting (Sec. 4.2). In each decoder layer, hierarchical cross-attention is followed by Swin self-attention over ray patch tokens. DPT decoder predicts the final image.

3 Background

RenderFormer Zeng et al. (2025) formulates rendering as a sequence-to-sequence prediction problem: a scene described by triangle meshes, materials, and emission is mapped to image-patch tokens and then to an HDR image. The network is trained across scenes and can render previously unseen scenes without per-scene optimization. Its two stages separate scene-wide light-transport modeling from image formation for a particular camera.

Triangle representation. Let N denote the number of scene triangles, including emissive triangles. Each triangle is represented by a feature token encoding its shading normals, reflectance parameters, and emission. Geometry enters attention through a spatial rotary positional encoding (RoPE) constructed from the three vertex positions. The scene sequence also includes R learned register tokens that provide additional capacity for global information.

View-independent stage. The first stage applies dense self-attention and feed-forward layers to the triangle and register tokens. Each triangle can aggregate information from the entire scene, producing a latent representation of light transport that includes interactions with distant surfaces and lights. We denote the resulting scene features by $Z^\star \in \mathbb{R}^{(N+R)\times D}$, where D is the feature width. Because this stage does not depend on the camera input, its output can be reused across views when geometry, materials, and lighting remain fixed.

View-dependent stage. The second stage initializes one query token for each 8×8 image patch from its bundle of camera-ray directions. Each decoder layer uses these patch states as queries and the fixed $Z^\star$ as the source of keys and values in dense cross-attention. Triangle positions are expressed in camera coordinates for spatial RoPE. Patch self-attention and feed-forward layers further process the retrieved features, and a Dense Prediction Transformer (DPT) Ranftl et al. (2021) head reconstructs the log-encoded HDR image.

Dense attention and surface discretization. The view-independent encoder evaluates $(N + R)^2$ attention pairs per layer and head, while the view-dependent decoder cross-attention evaluates $P(N + R)$. Their attention arithmetic therefore scales as $O(N^2D)$ and $O(PND)$ for fixed R, limiting the model's scalability to higher triangle counts.

4 METHOD

Our method is a two-stage neural rendering pipeline built on reweighted hierarchical attention. It first encodes scene-wide light transport through coarse and fine surface representations, then uses view-dependent queries to synthesize an image from a target camera. Our pipeline is illustrated in Fig. 2. We first describe the surface hierarchy (Sec. 4.1), then present reweighted hierarchical attentions including surface area reweighting and distinct hierarchical attention for scene encoding and view-dependent feature aggregation (Sec. 4.2). Finally, we describe training details (Sec. 4.3).

4.1 SURFACE HIERARCHY

We retain RenderFormer's triangle embeddings and learned register tokens (Sec. 3), and build a coarse surface hierarchy over the non-emissive surface triangles, denoted by $\mathcal{T}$.

A 3D token hierarchy aggregates spatially related surface features into regional representations, allowing each coarse token to preserve the distinct spatial relevance of the region it represents. Such hierarchies commonly combine spatial grouping with sequence serialization to organize irregular 3D data into token sequences suited to efficient attention computation Wu et al. (2024); Xu et al. (2026). However, directly splitting a scene-wide Hilbert sequence into groups containing equal numbers of triangles can place spatially distant surfaces in the same group (Fig. 3). We therefore form clusters within each mesh component before ordering them globally. This spatially cohesive grouping helps reduce information loss when pooling triangle features into coarse tokens. As attention depends strongly on spatial location, keeping distant surfaces in separate clusters also helps preserve their distinct relevance to a query.

Component-constrained surface partition. Within each component, we order triangle centroids along a Hilbert curve following Point Transformer V3 Wu et al. (2024). We partition this sequence into disjoint clusters with comparable surface areas to reduce sensitivity to tessellation density. This grouping also helps keep coarse tokens spatially localized. Pooling distant surfaces into a single token can obscure their distinct relevance to a query, distorting the spatial distribution of attention.

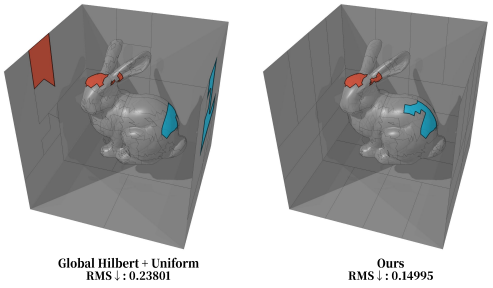

Figure 3: **Spatial locality of surface clusters.** Splitting a scene-wide Hilbert sequence into groups with equal triangle counts (left) can assign distant triangles to the same cluster. Our component-constrained partition (right) keeps these surfaces in separate clusters.

Cluster-centroid Hilbert ordering. To reduce spatial jumps in the triangle sequence at component boundaries, we again apply Hilbert ordering, this time to the area-weighted centroids of all clusters in the scene. We form the final triangle sequence by concatenating the triangle lists of the ordered clusters, preserving the order within each cluster.

4.2 REWEIGHTED HIERARCHICAL ATTENTION

Surface-area reweighting. The same surface geometry can be represented by different triangulations. For a fixed query, we observe that newly tessellated triangles retain nearly the same query–key logits as their parent triangle under naive attention. The subdivided region therefore receives more attention after softmax, even though the scene is unchanged. One possible reason is that learning tessellation-independent behavior implicitly from geometric cues encoded by RoPE is itself difficult, and may additionally require tessellation-augmented training examples.

This sensitivity can be viewed as a non-uniform sampling problem: more densely tessellated regions contribute more tokens per unit surface area. Continuous-attention formulations Calvello et al. (2025); Berner et al. (2026) account for such sampling differences by expressing attention as an integral and weighting its discrete approximation according to the represented measure. Consider

a query q and key/value fields $k(x), v(x)$ on a surface $\mathcal{M}$. With score $s(q,x) = q^\top k(x)/\sqrt{d}$ and head width d, a triangle partition with positive areas A_i gives

$$\begin{aligned}\mathcal{A}_{\mathcal{M}}(q) &= \frac{\int_{\mathcal{M}} e^{s(q,x)} v(x)\,\mathrm{d}A(x)}{\int_{\mathcal{M}} e^{s(q,x)}\,\mathrm{d}A(x)} \\ &\approx \frac{\sum_{i\in\mathcal{T}} A_i e^{s_i} v_i}{\sum_{i\in\mathcal{T}} A_i e^{s_i}} \\ &= \sum_{i\in\mathcal{T}} \mathrm{softmax}_i(s_i + \log A_i) v_i.\end{aligned} \tag{1}$$

Here $s_i = s(q,\xi_i)$ and $v_i = v(\xi_i)$ are evaluated at a point ξ_i on triangle i. Inspired by this formulation, we apply the resulting log-area bias to triangle and cluster keys according to the surface area they represent. As shown in Fig. 4, the reweighted model better preserves the attention distribution across the scene after tessellation. Register tokens have no associated surface area, while the area of a light token has no meaningful interpretation in the current model. We therefore use zero key bias for both. Large area lights can be approximated by subdividing them into multiple light tokens.

Global and local scene encoding. We use a view-independent encoder to model global light transport across the scene. Dense attention over N triangle tokens incurs $O(N^2)$ computational cost. To handle long sequences efficiently, prior work combines sliding-window attention with global tokens to retain long-range context Beltagy et al. (2020); Zeng et al. (2026). We follow this local–global design: each triangle attends jointly to a bidirectional window in the ordered sequence and all cluster, light, and register tokens through a single area-reweighted softmax. However, cluster keys and values are regenerated at each layer by area-weighted pooling of their constituent triangles' projected keys and values after applying RoPE Zhu et al. (2021). Directly pooling triangle features avoids maintaining cluster states across layers, but can lose feature variation within each cluster. We adapt the unnormalized attention-error proxy of SVG-EAR Zhou et al. (2026a) to a surface-area measure. For a fixed query and head, let $A_g = \sum_{i\in g} A_i$ and $w_i = A_i/A_g$ for cluster g. Area-weighted key pooling gives the coarse content logit $\mu_g = \sum_{i\in g} w_i s_i$, where $s_i = q^\top k_i/\sqrt{d}$ excludes the area bias. The fine and coarse attention logits are therefore $s_i + \log A_i$ and $\mu_g + \log A_g$, respectively. Dividing their exponentials by the represented areas gives the corresponding unnormalized attention densities. With $\delta_i = s_i - \mu_g$, we define the area-averaged squared density discrepancy as

$$\begin{aligned}r_g^{(A)} &= \frac{1}{A_g}\sum_{i\in g} A_i \left(\frac{e^{s_i+\log A_i}}{A_i} - \frac{e^{\mu_g+\log A_g}}{A_g}\right)^2 \\ &= \sum_{i\in g} w_i \left(e^{s_i} - e^{\mu_g}\right)^2 \\ &= e^{2\mu_g}\sum_{i\in g} w_i \left(e^{\delta_i} - 1\right)^2 \\ &\approx e^{2\mu_g}\,\mathrm{Var}_w(s).\end{aligned} \tag{2}$$

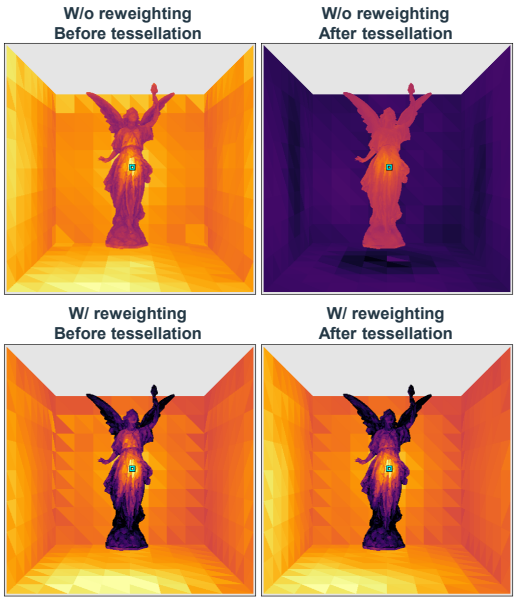

Figure 4: **Reduced sensitivity to tessellation with surface-area reweighting.** We compare the models with (bottom) and without (top) surface-area reweighting, before (left) and after (right) subdividing Lucy into 16× as many triangles. The cyan box marks the same query patch in all panels. Since child triangles from the same parent have similar value features, we sum their attention probabilities and display the total on the original parent triangle. Colors show these probabilities averaged over all decoder layers on a shared logarithmic scale, ranging from dark to bright. Tessellation shifts the model's attention toward Lucy, while the reweighted model better preserves the distribution of attention across the scene.

Here $\mathrm{Var}_w(s) = \sum_{i \in g} w_i \delta_i^2$ is the area-weighted logit variance. The last step uses $\sum_{i \in g} w_i \delta_i = 0$ and $e^{\delta_i} - 1 \approx \delta_i$ for small within-cluster logit variation. The proxy couples coarse attention density with within-cluster logit variation. Our spatially cohesive hierarchy helps limit this variation, while sliding-window attention retains direct access to fine local features.

Query-dependent hierarchical feature aggregation. The view-dependent decoder uses hierarchical cross-attention to aggregate scene features into patch features for DPT to decode into pixel colors. Following the coarse-guided selection principle of LLSA Zhou et al. (2026b), we use coarse attention to identify regions for fine-level computation. Each patch query selects the union of clusters containing triangles directly visible in the patch and the top-k clusters ranked by area-reweighted attention scores. Each selected cluster token is replaced by all its constituent triangle tokens, exposing fine surface detail while the remaining cluster tokens provide global context. The cross-attention context also retains register and light tokens. The ranked cluster probabilities (Fig. 5, top left) fall steeply, so a small top-k captures the leading high-weight clusters. We further validate this selective expansion experimentally (Fig. 5, top right): top-k expansion with the remaining coarse context yields final patch features with high cosine similarity to dense Stage 2 inference (0.966 at $k = 8$). Furthermore, the visual comparison (bottom) also shows our results with more faithful shadows and reflections, possibly because the reweighted attention on retrieved fine tokens. In each decoder layer, hierarchical cross-attention is followed by Swin self-attention Liu et al. (2021), which exchanges information among neighboring image patches efficiently and has been shown to be effective for resolution generalization.

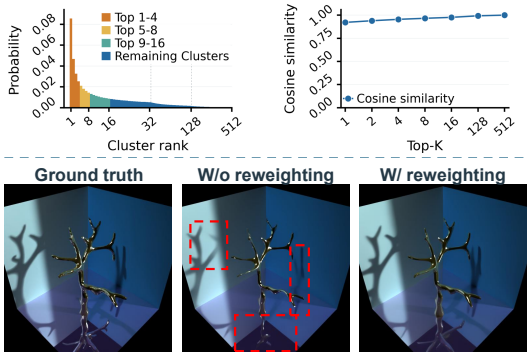

Figure 5: **Surface-area reweighting for hierarchical attention. Top left:** Ranked cluster attention probabilities. **Top right:** Mean cosine similarity between final patch features from top-k and dense Stage 2 inference. **Bottom:** We compare ground truth (left), the models with (right) and without (middle) reweighting under Stage 2 top-8 cluster expansion setup. The reweighting model produces more precise shadows and reflections.

4.3 Training

We initialize from the released RenderFormer Zeng et al. (2025) `v1.1-swin-large` weights and fine-tune all model components jointly. Our training set contains approximately 125k scenes and 500k images at 512×512, generated using the same procedure as RenderFormer. Each scene contains about 4k to 10k triangles. We train for 1 epoch using BF16 mixed precision and AdamW Loshchilov & Hutter (2019) with a learning rate of 5×10^{-5}. The training objective combines a log-L1 loss on linear-HDR predictions with LPIPS Zhang et al. (2018) on tone-mapped images:

$$\mathcal{L} = \mathcal{L}_{\mathrm{logL1}} + 0.05\, \mathcal{L}_{\mathrm{LPIPS}}. \tag{3}$$

The model uses feature width $D = 1024$ and $R = 32$ register tokens. The 512×512 images are divided into 8×8 patches, giving $P = 4096$ patch tokens. We use weight decay 0.01, 2000 warmup steps followed by cosine decay, gradient accumulation two, and gradient clipping at norm 1.0.

5 Results

5.1 Comparisons

We primarily compare our method with RenderFormer Zeng et al. (2025) and RenderFormer-V2 Zeng et al. (2026). As our work focuses on scaling transformer-based neural rendering to higher triangle counts, we construct 75 test scenes containing 500 to one million triangles. These scenes follow the assembly procedure used for training, primarily combining objects selected from TexVerse Zhang et al. (2025).

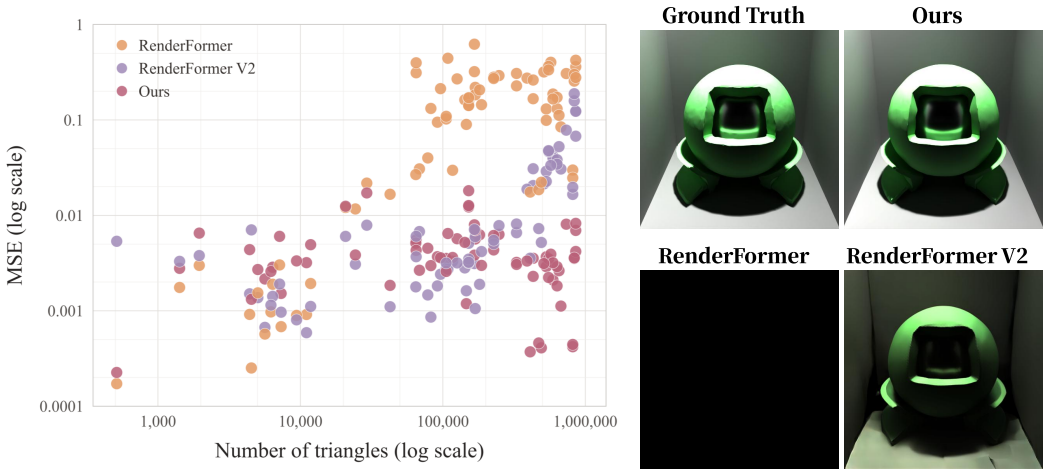

Figure 6: **Comparison with RenderFormer and RenderFormer-V2. Left:** MSE versus triangle count for RenderFormer, RenderFormer-V2, and our method on 75 test scenes containing 500 to one million triangles. Both axes use logarithmic scales; lower MSE is better. **Right:** Qualitative comparison on a tessellated scene containing 673k triangles, illustrating our method's ability to maintain rendering quality at large triangle counts.

As shown in Fig. 6 (left), RenderFormer performs well on scenes with fewer than approximately $10k$ triangles, but its error (reported as mean square error) increases markedly at larger scales. Its outputs can exhibit pronounced darkening and distorted surface appearance (Fig. 6, right), consistent with the attention drift discussed earlier. Concurrent work RenderFormer-V2 mainly extends the scene representation to heterogeneous primitives and improves scalability over RenderFormer, but also shows quality degradation on scenes exceeding 100k triangles. Our method maintains consistently high rendering quality across the tested range, with low errors even on scenes approaching one million triangles. We provide a qualitative comparison on a tessellated example scene in Fig. 6 (right) to illustrate our method's advantage in scaling to larger triangle counts.

5.2 Ablation Study

We conduct ablation studies to assess the contribution of each component of our method. These experiments use a set of scenes containing an average of 100k triangles per scene. We evaluate rendering quality using PSNR, SSIM Wang et al. (2004), and LPIPS Zhang et al. (2018) on tone-mapped images, and HDR-FLIP Andersson et al. (2021) on linear HDR images. We further report inference time, measured on an NVIDIA L20 GPU, to assess computational efficiency.

Tab. 1 compares the trade-offs among applying reweighted hierarchical attention in one or both stages, using different cluster counts and grouping strategies, and applying surface-area reweighting. Hierarchical attention improves inference efficiency by aggregating fine features into coarse representations, at the cost of potential information loss. Replacing hierarchical attention with dense attention may slightly improve rendering quality but increasing inference time (Fig. 7). Increasing the number of clusters tends to produce smaller, more spatially cohesive groups, reducing the information lost when each group is represented by a coarse token. We also evaluate *Uniform clustering*, which orders all triangles in the scene along a Hilbert curve and splits the resulting sequence into groups with equal triangle counts. This alternative degrades all reported quality metrics relative to our grouping, supporting the benefit of spatially cohesive, area-constrained clusters. Removing surface-area reweighting causes a substantial quality drop. This highlights the importance of combining hierarchical context with surface-area reweighting. The dense variant with area bias alone offers some quality gains but incurs substantially higher inference cost. Our full pipeline achieves a favorable balance between rendering quality and inference efficiency. We provide more results and details in supplementary materials.

Table 1: Ablation study of area bias, hierarchical stage-1, LLCA stage-2, cluster budgets, and uniform clustering on the evaluation set. **Best** results are shown in bold and second-ranked results are underlined; ties share the same competition rank.

Variant	PSNR ↑	SSIM ↑	LPIPS ↓	HDR-FLIP ↓	Time(s) ↓
W/o hier. stage-1	24.36	0.9131	0.0713	0.1970	10.75
W/o hier. stage-2	24.57	0.9167	**0.0639**	0.1931	1.71
Cluster # = 256	23.97	0.9073	0.0792	0.1998	**0.98**
Cluster # = 1024	24.83	0.9169	0.0655	0.1938	1.27
Uniform clustering	23.93	0.9109	0.0745	0.1977	1.16
W/o reweighting	17.66	0.8558	0.1709	0.2549	1.28
W/ reweighting only	**25.02**	**0.9225**	0.0887	**0.1724**	11.01
Ours (full)	24.50	0.9138	0.0700	0.1955	1.01

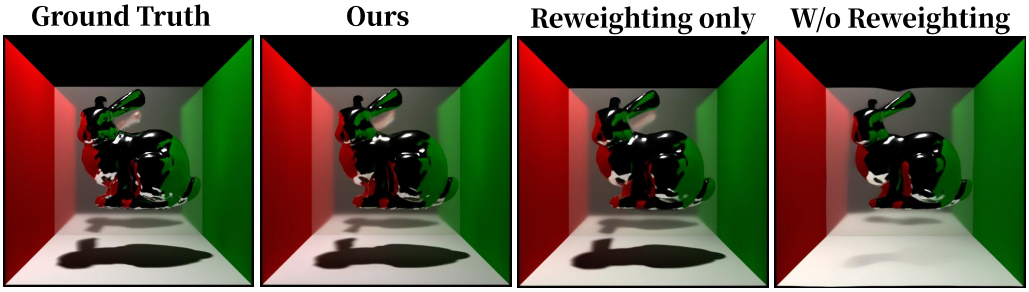

Figure 7: **Visual comparisons for ablation studies.**

6 Conclusion

We presented TriRender, a neural rendering pipeline for scenes with high triangle counts based on reweighted hierarchical attention. Surface-area reweighting accounts for the area represented by both triangle and cluster tokens. A component-constrained Hilbert hierarchy groups spatially related triangles into comparable area clusters. The scene encoder combines global context with sliding-window attention, while the view-dependent decoder expands clusters selected by direct visibility and top-k retrieval, retaining the remaining clusters as coarse context. This concentrates fine interactions on selected surface regions while preserving access to scene-wide information.

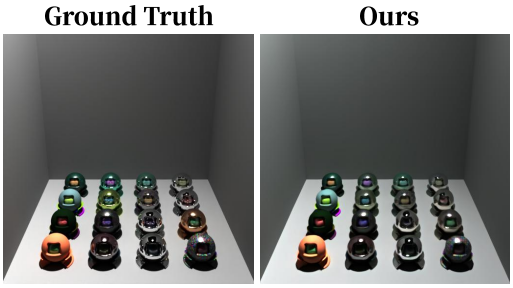

Figure 8: **Limitation.** TriRender may lose appearance details on very small triangles.

TriRender has several limitations. Our method may produce blurred results when rendering very small triangles with random textures (Fig. 8). Increasing the output image resolution may help alleviate this issue. For scenes with few triangles, our hierarchical attention may incur unnecessary overhead, which can be avoided by switching to dense attention. Our focus is on scaling to higher triangle counts and accelerating inference within a triangle-mesh representation. We have therefore not extended the model to the broader range of scene primitives supported by RenderFormer-V2 Zeng et al. (2026), such as participating media and environment maps. Adapting both our hierarchy and reweighting scheme to heterogeneous scene primitives remains a direction for future work.

AI USE STATEMENT

(This section is **required** and does not count toward the page limit.)

In this work, we used generative AI tools for *design or provide feedback on research methodology or experiments, implement methods, assist with translation*. We have not used generative AI tools for *clean and reformat dataset, support qualitative and thematic data analysis, interpret results*, and *Generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses* are not applicable to this work. Additionally, we used generative AI tools for *create or edit software code, summarize or analyze existing literature, sourcing/searching for information, edit a research paper to improve readability, identify relevant literature*. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

See the ICLR 2027 AI Policy for Authors for more details. This statement should not be more than 1 page.

REFERENCES

Pontus Andersson, Jim Nilsson, Peter Shirley, and Tomas Akenine-Möller. Visualizing errors in rendered high dynamic range images. In *Eurographics 2021 - Short Papers*. The Eurographics Association, 2021. doi: 10.2312/egs.20211015. URL https://diglib.eg.org/items/a74ea1e0-b1e3-4231-b118-46ad9a18e85b.

Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

Julius Berner, Miguel Liu-Schiaffini, Jean Kossaifi, Valentin Duruisseaux, Boris Bonev, Kamyar Azizzadenesheli, and Anima Anandkumar. Principled approaches for extending neural architectures to function spaces for operator learning. *Nature Machine Intelligence*, 8:1173–1181, 2026. doi: 10.1038/s42256-026-01267-z. URL https://www.nature.com/articles/s42256-026-01267-z.

Edoardo Calvello, Nikola B. Kovachki, Matthew E. Levine, and Andrew M. Stuart. Continuum attention for neural operators. *Journal of Machine Learning Research*, 26(300):1–52, 2025. URL https://jmlr.org/papers/v26/24-0879.html.

Pedro Hermosilla, Sebastian Maisch, Tobias Ritschel, and Timo Ropinski. Deep-learning the latent space of light transport. In *Computer Graphics Forum*, volume 38, pp. 207–217. Wiley Online Library, 2019.

David Hilbert. Ueber die stetige Abbildung einer Linie auf ein Flächenstück. *Mathematische Annalen*, 38:459–460, 1891. doi: 10.1007/BF01199431. URL https://doi.org/10.1007/BF01199431.

Ruofan Liang, Zan Gojcic, Huan Ling, Jacob Munkberg, Jon Hasselgren, Chih-Hao Lin, Jun Gao, Alexander Keller, Nandita Vijaykumar, Sanja Fidler, et al. Diffusionrenderer: Neural inverse and forward rendering with video diffusion models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26069–26080. IEEE, 2025.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *2021 IEEE/CVF international conference on computer vision (ICCV)*, pp. 9992–10002. Ieee, 2021.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Morgan McGuire and Michael Mara. Efficient gpu screen-space ray tracing. *Journal of Computer Graphics Techniques (JCGT)*, 3(4):73–85, 2014. URL https://jcgt.org/published/0003/04/04/.

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 *ECCV*, 2020.

Oliver Nalbach, Elena Arabadzhiyska, Dushyant Mehta, H-P Seidel, and Tobias Ritschel. Deep shading: convolutional neural networks for screen space shading. In *Computer graphics forum*, volume 36, pp. 65–78. Wiley Online Library, 2017.

Nithin Raghavan, Yan Xiao, Kai-En Lin, Tiancheng Sun, Sai Bi, Zexiang Xu, Tzu-Mao Li, and Ravi Ramamoorthi. Neural free-viewpoint relighting for glossy indirect illumination. In *Computer Graphics Forum*, volume 42, pp. e14885. Wiley Online Library, 2023.

Gilles Rainer, Adrien Bousseau, Tobias Ritschel, and George Drettakis. Neural precomputed radiance transfer. In *Computer graphics forum*, volume 41, pp. 365–378. Wiley Online Library, 2022.

René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Int. Conf. Comput. Vis.*, pp. 12179–12188, 2021. URL https://openaccess.thecvf.com/content/ICCV2021/html/Ranftl_Vision_Transformers_for_Dense_Prediction_ICCV_2021_paper.html.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient transformers: A survey. *ACM Computing Surveys*, 55(6):1–28, 2022.

Ayush Tewari, Justus Thies, Ben Mildenhall, Pratul Srinivasan, Edgar Tretschk, Wang Yifan, Christoph Lassner, Vincent Sitzmann, Ricardo Martin-Brualla, Stephen Lombardi, et al. Advances in neural rendering. In *Computer Graphics Forum*, volume 41, pp. 703–735. Wiley Online Library, 2022.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004. URL https://ece.uwaterloo.ca/~z70wang/research/ssim/.

Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point Transformer V3: Simpler, faster, stronger. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2024. URL https://arxiv.org/abs/2312.10035.

Hanggao Xin, Shaokun Zheng, Kun Xu, and Ling-Qi Yan. Lightweight bilateral convolutional neural networks for interactive single-bounce diffuse indirect illumination. *IEEE transactions on visualization and computer graphics*, 28(4):1824–1834, 2020.

Bing Xu, Mukund Varma T, Cheng Wang, Tzu-Mao Li, Lifan Wu, Bartlomiej Wronski, Ravi Ramamoorthi, and Marco Salvi. A generalizable light transport 3D embedding for global illumination. In *ACM SIGGRAPH 2026 Conference Papers*, 2026. doi: 10.1145/3799902.3811095. URL https://arxiv.org/abs/2510.18189v2.

Jianwei Yang, Chunyuan Li, Pengchuan Zhang, Xiyang Dai, Bin Xiao, Lu Yuan, and Jianfeng Gao. Focal attention for long-range interactions in vision transformers. *Advances in neural information processing systems*, 34:30008–30022, 2021.

Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, Yuxing Wei, Lean Wang, Zhiping Xiao, et al. Native sparse attention: Hardware-aligned and natively trainable sparse attention. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23078–23097, 2025.

Chong Zeng, Yue Dong, Pieter Peers, Hongzhi Wu, and Xin Tong. Renderformer: Transformer-based neural rendering of triangle meshes with global illumination. In *ACM SIGGRAPH 2025 Conference Papers*, pp. 1–11, 2025.

Chong Zeng, Yue Dong, Pieter Peers, Lvmin Zhang, and Maneesh Agrawala. Renderformer-v2: Neural rendering with heterogeneous scene primitives. In *Eur. Conf. Comput. Vis.*, 2026.

Zheng Zeng, Valentin Deschaintre, Iliyan Georgiev, Yannick Hold-Geoffroy, Yiwei Hu, Fujun Luan, Ling-Qi Yan, and Miloš Hašan. Rgb↔ x: Image decomposition and synthesis using material-and lighting-aware diffusion models. In *ACM SIGGRAPH 2024 conference papers*, pp. 1–11, 2024.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018.

Yibo Zhang, Li Zhang, Rui Ma, and Nan Cao. TexVerse: A universe of 3D objects with high-resolution textures. *arXiv preprint arXiv:2508.10868*, 2025. URL https://arxiv.org/abs/2508.10868.

Chuankun Zheng, Yuchi Huo, Hongxiang Huang, Hongtao Sheng, Junrong Huang, Rui Tang, Hao Zhu, Rui Wang, and Hujun Bao. Neural global illumination via superposed deformable feature fields. In *SIGGRAPH Asia 2024 Conference Papers*, pp. 1–11, 2024.

Xuanyi Zhou, Qiuyang Mang, Shuo Yang, Haocheng Xi, Jintao Zhang, Huanzhi Mao, Joseph E. Gonzalez, Kurt Keutzer, Ion Stoica, and Alvin Cheung. SVG-EAR: Parameter-free linear compensation for sparse video generation via error-aware routing. *arXiv preprint arXiv:2603.08982*, 2026a. URL https://arxiv.org/abs/2603.08982.

Yifan Zhou, Zeqi Xiao, Tianyi Wei, Shuai Yang, and Xingang Pan. Trainable log-linear sparse attention for efficient diffusion transformers. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 9424–9433, June 2026b.

Chen Zhu, Wei Ping, Chaowei Xiao, Mohammad Shoeybi, Tom Goldstein, Anima Anandkumar, and Bryan Catanzaro. Long-Short Transformer: Efficient transformers for language and vision. In *Advances in Neural Information Processing Systems*, volume 34, 2021. URL https://papers.nips.cc/paper_files/paper/2021/hash/9425be43ba92c2b4454ca7bf602efad8-Abstract.html.

Zhenhai Zhu and Radu Soricut. H-transformer-1d: Fast one-dimensional hierarchical attention for sequences. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 3801–3815, 2021.