A Generalizable Light Transport 3D Embedding for Global Illumination

BING XU, UC San Diego and NVIDIA, USA
MUKUND VARMA T, UC San Diego, USA
CHENG WANG, UC San Diego, USA
TZU-MAO LI, UC San Diego, USA
LIFAN WU, NVIDIA, USA
BARTLOMIEJ WRONSKI, NVIDIA, USA
RAVI RAMAMOORTHI, UC San Diego, USA
MARCO SALVI, NVIDIA, USA

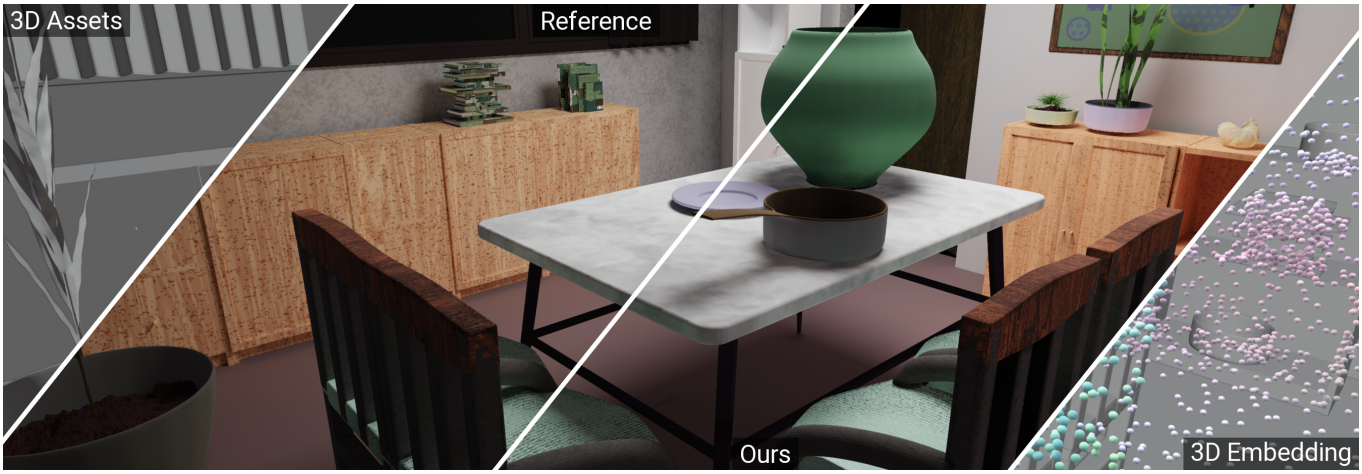


Fig. 1. Our scalable transformer-based, generalizable light-transport model takes 3D scene assets (leftmost slice)—geometry, materials, and lighting as a point cloud—and encodes global light transport into a 3D embedding (rightmost slice) via latent codes anchored at scene points. The model is view- and resolution-independent. Center slices: reference vs. our predicted global illumination for an unseen scene at 2688×1152 resolution (cropped for space).

Global illumination (GI) is essential for realism but remains computationally expensive. While per-scene neural methods lack generalization and screen-space approaches inherently suffer from view inconsistency, prior 3D neural rendering methods face a severe scalability barrier, restricting them to small, object-centric meshes. To overcome these trade-offs, we introduce a generalizable light transport 3D embedding that predicts global illumination directly from 3D scene configurations without rasterized or path-traced illumination cues, per-scene retraining or screen-space limitations. We employ a point-based representation to decouple our embedding from the original scene topology, then utilize a linear-complexity transformer to encode long-range light transport. This design scales to environments with millions of triangles, enabling the first generalizable GI learning on complex, high-fidelity indoor scenes, far beyond prior limits. To achieve this, we enforce a local query mechanism where rendering queries are processed independently under 3D supervision. This ensures constant complexity per pixel relative to scene size, yielding view-consistent and resolution-agnostic rendering without the memory bottlenecks typical of globally coupled attention. We further demonstrate versatility by re-targeting the encoder with limited fine-tuning, presenting preliminary results on spatial-directional radiance field prediction for glossy materials and validating transfer to downstream rendering tasks.

Authors' addresses: Bing Xu, binxu@nvidia.com, UC San Diego and NVIDIA, USA; Mukund Varma T, tmukund@ucsd.edu, UC San Diego, USA; Cheng Wang, chengwang@ucsd.edu, UC San Diego, USA; Tzu-Mao Li, tzli@ucsd.edu, UC San Diego, USA; Lifan Wu, lifanw@nvidia.com, NVIDIA, USA; Bartlomiej Wronski, elirian@gmail.com, NVIDIA, USA; Ravi Ramamoorthi, ravir@cs.ucsd.edu, UC San Diego, USA; Marco Salvi, msalvi@nvidia.com, NVIDIA, USA.

SIGGRAPH Conference Papers '26, July 19–23, 2026, Los Angeles, CA, USA

ACM ISBN 979-8-4007-2554-8/2026/07
https://doi.org/10.1145/3799902.3811095

1 INTRODUCTION

A long-term goal of computer graphics has been to efficiently simulate complex light-matter interactions. While researchers have derived mathematical models for physical phenomena [Christensen et al. 2016], recent years have witnessed a different paradigm: learning data-driven priors to approximate these processes [OpenAI 2024]. With the growing availability of 3D data [Siddiqui et al. 2024], we seek to answer a central question: can data priors alone be used to train a generalizable model that predicts 3D light transport directly from scene configurations, partially bypassing explicit simulation?

Monte Carlo rendering remains the workhorse of photorealism but is computationally intensive. Recent neural global illumination methods [Diolatzis et al. 2022; Granskog et al. 2020; Rainer et al. 2022;

Ren et al. 2013; Zheng et al. 2023] approximate light transport to enable efficient reuse of lighting computation, but they typically overfit to individual scenes, preventing generalization. Conversely, cross-scene generalization efforts, including neural denoisers [Áfra 2024] or G-buffer–based predictors [Nalbach et al. 2017], often operate in 2D screen space. These methods suffer from view-inconsistency and fail to capture global illumination from off-screen geometry.

Closer to our goal are methods that learn light transport directly in 3D. However, these approaches face a severe **scalability barrier**. Pioneering work by Hermosilla et al. [2019] based on convolutions struggles to model long-range global interactions, and the recent transformer-based approaches like RenderFormer [Zeng et al. 2025] rely on global attention with quadratic complexity. This computational bottleneck restricts them to simple, object-centric meshes, making them well-suited to object-centric setups but unsuitable for the dense geometry of real-world environments.

In this paper, we introduce a generalizable light transport 3D embedding designed to prioritize scalability and versatility. We achieve a scalable architecture by reducing computational complexity in three key ways: 1) *Point-based primitives* decouple lighting resolution from the original scene topology and unify light sources with geometry into a single representation. 2) *A linear-complexity transformer* replaces quadratic global attention to efficiently encode long-range light transport into the 3D embedding. 3) *Independent local decoding*. We retrieve features only from a fixed local neighborhood (k-NN) of the embedding regardless of the scene size and process each query independently. This ensures no dependencies exist between separate rendering rays, enabling flexible training batch sizes and resolution-agnostic inference. Finally, we use direct 3D supervision in world space rather than 2D image space. This guarantees that our learned embedding is inherently view-consistent and robust to camera trajectory shifts.

In summary, we make the following contributions:

- A scalable 3D embedding using point clouds and linear-complexity transformers to overcome quadratic memory bottlenecks, enabling global transport modeling on millions of triangles (§3.2,§3.3).
- Resolution-agnostic decoding: A local decoding mechanism with constant per-pixel complexity that, via 3D supervision, ensures view-consistent and generalizable GI for large-scale scenes (§3.4).
- We curate and release a ∼14k complex indoor scenes dataset with diverse layouts, geometries, and textures, serving as a benchmark for learning light transport (§4).
- Architectural versatility: A transferable encoder capable of supporting dedicated tasks, demonstrated by extending our primary diffuse GI model to predict spatial-directional radiance fields for glossy materials (§5.3).

2 RELATED WORK

We focus our discussion on data-driven methods, particularly those leveraging neural networks to approximate light transport or operate directly on 3D data.

Per-scene neural rendering. Classical Precomputed Radiance Transfer (PRT) [Ramamoorthi et al. 2009; Sloan et al. 2002] and its neural extensions [Raghavan et al. 2023; Rainer et al. 2022] enable efficient GI but rely on per-scene precomputation. Similarly, early neural regressions [Ren et al. 2013] and subsequent scene representations have focused on specific dynamic factors, such as moving objects [Zheng et al. 2024, 2023], variable lighting or materials [Diolatzis et al. 2022; Gao et al. 2023], and changing viewpoints [Eslami et al. 2018; Granskog et al. 2020]. Hybrid neural renderers incorporate 3D scene reasoning via point-based light transport modules, trained per scene, while synthesizing final images in screen space [Sanzenbacher et al. 2020]. However, these methods remain fundamentally instance-specific. In contrast, our approach targets *scene-level generalization*, learning a light transport embedding directly from raw 3D configurations without per-scene training.

Generalizable screen-space techniques. Neural denoising and deep shading are two major approaches that aim for scene-level generalization. Denoising methods [Bako et al. 2017; Chaitanya et al. 2017; Gharbi et al. 2019; Vogels et al. 2018; Xu et al. 2019] operate as post-processing, while deep shading [Nalbach et al. 2017; Xin et al. 2022] infers shading from G-buffers. While efficient, their reliance on screen-space input leads to view inconsistencies and misses off-screen light transport effects.

Learning light transport in 3D. A more robust alternative is to learn light transport directly in the 3D world space. Hermosilla et al. [2019] pioneered this direction but relied on convolutions, which generally prioritize local features and struggle to model long-range global interactions. More recently, Zeng et al. [2025] successfully applied global transformer architectures to mesh vertices, producing high-fidelity results for complex glossy and specular effects. While effective for single objects, their reliance on standard global attention introduces a quadratic computational bottleneck. This complexity constrains the approach to low-polygon scenes (e.g., ≈4k triangles), and is not the natural fit for dense, complex environments (see Figure 2). Furthermore, our design adopts decoupled geometry-ray decoding, where each query independently attends to a local neighborhood of scene primitives, yielding view consistency and resolution flexibility. We provide a detailed structural comparison with [Zeng et al. 2025] in §5.5 and related statistics in the Supplementary §3. Looking ahead, a shared challenge for both methods is to further scale up with rendering parameters to capture the full spectrum of light transport effects.

Deep learning on irregular geometric data. Processing irregular 3D data requires handling permutation invariance and varying density. Early MLP-based approaches [Qi et al. 2017a,b] and subsequent graph [Wang et al. 2019] or continuous convolution methods [Li et al. 2019; Thomas et al. 2019] proved effective for local geometry. Attention-driven architectures such as Point Transformer [Zhao et al. 2021] further enhance the integration of global context. PointTransformerV3 [Wu et al. 2024] (PTV3) replaces per-layer KNN with serialization-based patch attention to scale to large point clouds; we adopt it as our encoder backbone. A brief background is provided in Supplementary §1.2. This choice prioritizes scalability and architectural simplicity, enabling efficient learning of complex relationships within 3D point clouds for global illumination.

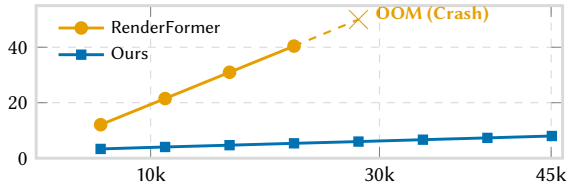


Fig. 2. Memory scalability. We benchmark *Peak Training VRAM (GB)* (y) with increasing *Number of primitives* (x) on a single scene using a tiny $\mathbf{64^2}$ **resolution**. Even with this reduced query count, RenderFormer rapidly exhausts a 48GB GPU. See more discussions in §5.4 and supplementary.

3 LIGHT TRANSPORT 3D EMBEDDING

We introduce a generalizable framework for predicting global illumination directly from 3D configurations. Inspired by the parallel between the light transport operator and attention (§3.1), we detail our pipeline's three key stages, as illustrated in Figure 4: Scene *Discretization* (§3.2), *Global Light Transport Encoding* (§3.3) where we encode these points into a high-dimensional neural representation ($\mathbf{F}_i$) that captures the light transport function of the given scene, and *Local Query Decoding* (§3.4).

3.1 Motivation: An Analogy Bridging Light Transport and Attention

Light transport is often expanded as a Neumann series [Arvo et al. 1994], where global illumination is resolved by recursively propagating energy between surface points. This mirrors the stacked self-attention operation in transformers [Vaswani et al. 2017]. Just as the transport operator T (Soler et al. [2022]) recursively simulates bounces, passing features through successive transformer layers allows the network to capture increasingly complex, multi-bounce interactions. We offer a visual intuition for this parallel in Figure 3, illustrating that attention weights naturally highlight scene regions contributing to a point's illumination.

Guided by this analogy, we design our Encoder to act as the global solver: by stacking deep attention layers, it "bakes" the multi-bounce simulation results into per-point embeddings. This allows our Decoder to simplify to a **local retrieval task**, avoiding the expensive global attention used in prior work [Zeng et al. 2025] and enabling resolution-independent inference.

3.2 Points as the Intermediate Scene Representation

Unlike prior generalizable methods that rely on fixed mesh topology (e.g., vertices) or view-dependent 2D projections [Nalbach et al. 2017], we adopt point clouds as our intermediate representation (IR), deriving per-point attributes from the 3D scene. This design offers key advantages: 1) scalability: point density can adjust based on importance; 2) generality: decoupled from original geometry, suitable for other sources like scanned data; and 3) simplicity: easier for neural networks to handle than meshes, with no connectivity.

Light sources. While most prior neural GI works focus on distant illumination (e.g., environment maps), this approximation fails to capture near-field effects. We instead use local area lights to produce rich, spatially varying illumination essential for indoor rendering. By representing light sources as point clouds, we unify the treatment

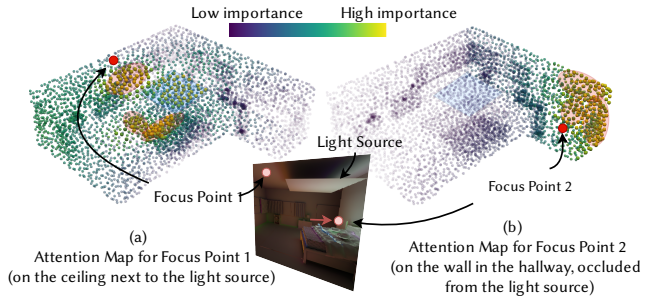


Fig. 3. Attention visualization for a bedroom with a hallway. We show attention heatmaps from our light transport encoder for two focus points. Points are shaded by importance, with the top 200 scores enlarged. The blue shaded square denotes the light source. In (a), we see four groups of scene points having the largest impact on the focus point (highlighted in blue square and orange clusters), which likely corresponds to 4 bounces of ray tracing. In (b), the focus point is on the wall of the hallway where the light source is occluded. This serves as a visual hint for our analogy between the light transport operator and the attention mechanism in §3.1.

of emissive and non-emissive geometries. This makes our method agnostic to the lighting type, effortlessly supporting area lights, point lights, or emissive strips without architectural changes.

Scene points. As a pre-processing step (Figure 4, left), we convert the scene into an intermediate representation by sampling object geometry into M ($\approx 20k$) *Scene points*. Each point i is defined as a tuple $(\mathbf{p}_i, \mathbf{n}_i, \mathbf{c}_i, \mathbf{e}_i)$, representing position, normal, albedo, and emissivity, respectively. The full scene is thus denoted as $\{(\mathbf{p}_i, \mathbf{n}_i, \mathbf{c}_i, \mathbf{e}_i)\}^M$. This approach aligns with prior light transport approximations that also leverage point cloud inputs [Hašan et al. 2006; Hermosilla et al. 2019]. Note that the mesh vertices used in Zeng et al. [2025] can be viewed as a specific realization of this representation. Further analysis of point sampling is provided in §5.5.

Query points. At render time, *Query points* correspond to ray-intersections for shading, while during training, they serve as locations for loss evaluation. Each point is defined by features $\{(\mathbf{p}_j, \mathbf{n}_j, \mathbf{c}_j)\}$. Crucially, rather than using view-dependent path tracing samples for supervision like many prior works (e.g. [Zeng et al. 2025], which is limited by fixed resolutions and views), we uniformly sample N ($\approx$ 2 million) points from the scene surfaces. This approach decouples training from specific camera angles, enabling view and resolution independence. Further analysis is in §5.5. *Note we differentiate* Scene points *from* Query points *throughout the paper.*

3.3 Global Light Transport Encoding

Our goal is to approximate global illumination effects directly from 3D scene inputs—geometries, materials, light sources—without relying on any rasterized or path-traced illumination cues. GI requires simulating the full complexity of light transport, including multiple bounces against various surfaces and interactions with various material properties (BSDFs), before reaching the camera. Our observation from §3.1 motivates us to leverage a transformer-based encoder (dubbed *Light Transport Encoder* in Figure 4) to derive per-point neural primitives that implicitly model the light transport operator. As illustrated in Figure 4, we encode these *Scene points*

into a high-dimensional neural representation ($\mathbf{F}_i$) that captures the light transport function of the given scene.

Given that a large number of points ($M \approx 20$K) is required to represent an entire scene with sufficient density, naïvely borrowing the original transformer architecture leads to prohibitive memory requirements. This is because self-attention scales quadratically, necessitating the storage of massive $M \times M$ attention maps. Rather, we leverage the state-of-the-art point cloud encoder PointTransformerV3 [Wu et al. 2024], to derive a latent representation for each point.

Our experiments indicate that naïvely feeding all input features into the transformer yields suboptimal results. We attribute this to the attention operation becoming increasingly sparse with a large number of input tokens, which complicates the modeling of complex interactions between scene points. Yet, aggressively reducing the number of input points can lead to information loss, as a large indoor scene cannot be adequately represented by a limited set of points.

Therefore, we derive an embedding from a sparser set of points, while also aggregating features from their immediate neighbors. The primary purpose of this Nearest Neighbor Embedding (NNE) is to pre-aggregate local features and increase embedding capacity before the transformer stage, rather than merely reducing the point count. This mitigates the attention operation's "learning overhead" associated with large token counts while still preserving information from the full point set. We found that this hierarchical aggregation outperforms a flat PointTransformerV3 baseline by providing a more enriched latent representation. Formally, we write the Nearest Neighbor Embedding (see Figure 5(a)) operation as follows:

Nearest Neighbor Embedding

$$\{\widetilde{\mathbf{X}}_l\}^m = \mathrm{FPS}(\{\mathbf{X}_i\}^M), \ \textit{where}\ \mathbf{X}_i = \mathcal{F}((\mathbf{p}_i, \mathbf{n}_i, \mathbf{c}_i, \mathbf{e}_i)) \quad (1)$$

$$\widehat{\mathbf{X}}_{l,k} = \mathrm{concat}\left(\mathbf{X}_k - \widetilde{\mathbf{X}}_l, \widetilde{\mathbf{X}}_l\right) \quad \forall \mathbf{X}_k \in \mathrm{KNN}^k_{\widetilde{\mathbf{X}}_l}(\{\mathbf{X}_i\}^M) \quad (2)$$

$$\widetilde{\mathbf{F}}_l = \max_k \mathcal{G}(\widehat{\mathbf{X}}_{l,k}) \quad (3)$$

The embedding process proceeds in three stages. Here in the first equation, $\mathcal{F}$ is a projection function that maps the raw input attributes of each point i to a latent feature $\mathbf{X}_i$. We then utilize Farthest Point Sampling (FPS) to subsample the original scene points $\{\mathbf{X}_i\}^M$ to m points $\{\widetilde{\mathbf{X}}_l\}^m$. In our implementation, FPS is applied twice as a feature pre-aggregation step, halving the point count with each iteration (resulting in $m = M/4$), though this reduction factor remains flexible and can be adjusted as needed. After this step, we obtain subsampled point latents $\widetilde{\mathbf{X}}_l$. Next, we perform local feature aggregation. $\mathrm{KNN}^k_{\widetilde{\mathbf{X}}_l}$ identifies the k nearest neighbors for each subsampled point $\mathbf{X}_l$ from the original scene, and the difference vectors between the KNN center and its neighbors are computed. These are then concatenated with each center $\widetilde{\mathbf{X}}_l$ using concat, processed by a second projection $\mathcal{G}$, and then max-pooled along the k neighbors to yield the final aggregated embedding $\widetilde{\mathbf{F}}_l$.

As mentioned, we model interactions between these scene points using Point Transformer V3 (PTV3, [Wu et al. 2024]) as our backbone:

$$\{\mathbf{F}_l\}^m = \mathrm{PTV3}(\{\widetilde{\mathbf{F}}_l\}^m), \quad (4)$$

where PTV3 operates on the derived scene embedding across all points $\{\widetilde{\mathbf{F}}_l\}^m$ to derive the final per-point latent $\{\mathbf{F}_l\}^m$ (dubbed *Light Transport Embedding*). While standard point encoders typically rely on expensive k-Nearest Neighbor (KNN) searches per layer, PTV3 introduces *point cloud serialization*. It orders points along a space-filling curve and computes attention within non-overlapping patches of consecutive points (using a patch size of 1024). This significantly reduces memory and latency; however, because attention is local within each patch, the spatial coverage per patch limits how much 3D context a single layer can directly reach. Our NNE pre-aggregation is specifically designed to complement this. With FPS applied twice (m=$M/4$) and k=32 neighbors per stage, each NNE-stage-1 point summarizes $\sim k$ raw points and each stage-2 point summarizes $\sim k$ stage-1 points, so each NNE point fed into the transformer accumulates context from up to $\sim k^2$=1024 raw scene points. A PTV3 patch of 1024 such "enriched" points therefore touches a massive 3D region (nearly 1/5 of the entire scene at M=20k) that no patch of 1024 raw points could reach in a single attention operation. See Supplementary §1.2 for an extended background. The overall operation in this stage is denoted as the Light Transport Encoder in Figure 5(a).

3.4 Local Query Decoding

Our framework adopts a modular design, allowing the light-transport embedding to be repurposed for various rendering tasks by simply swapping task-specific decoders. We show that, after training on a large-scale indoor dataset of diffuse scenes for irradiance prediction (§5.2), the model can be fine-tuned to handle glossy reflections with limited additional training on a small glossy dataset, leveraging the pretrained weights (§5.3).

Given the Light Transport Embeddings $\{\mathbf{F}_l\}^m$ of a scene, we wish to estimate global illumination quantities for arbitrary 3D points. For view-independent training, we sample the scene; for rendering, we obtain the ray intersection points, which we define as *query points* (see §3.2 and Figure 5). To do so, we propose a *Local Query Decoder* that aggregates information from neighboring scene embeddings to synthesize the final desired output (e.g., irradiance).

Consistent with the previous section, we project each query point j characterized by its position, normal, and material properties, i.e. $(\mathbf{p}_j, \mathbf{n}_j, \mathbf{c}_j)$, into a latent feature $\widetilde{\mathbf{G}}_j$ using an MLP $\mathcal{H}$ (as in the equation below). Simultaneously, we retrieve the κ nearest neighbors from the scene embeddings $\{\mathbf{F}_l\}^m$, denoted as $\mathbf{KV}$. Since neighbor importance varies based on distance and occlusion, simple averaging is insufficient; we therefore employ a learned aggregation strategy. We verify this experimentally in §5.5. To encode the spatial relationship, we compute the relative distance vector $\Delta\mathbf{p}_{j\kappa}$ between the query and each neighbor, mapping it to a high-dimensional positional encoding $\mathbf{P}_{j\kappa}$ via γ:

$$\widetilde{\mathbf{G}}_j = \mathcal{H}(\mathbf{p}_j, \mathbf{n}_j, \mathbf{c}_j) \quad (5)$$

$$\mathbf{KV} = \mathrm{KNN}^\kappa_{\widetilde{\mathbf{G}}_j}(\{\mathbf{F}_l\}^m) \quad (6)$$

$$\mathbf{P}_{j\kappa} = \gamma(\Delta\mathbf{p}_{j\kappa}), \Delta\mathbf{p}_{j\kappa} = \mathbf{p}_j - \mathbf{p}_\kappa. \quad (7)$$

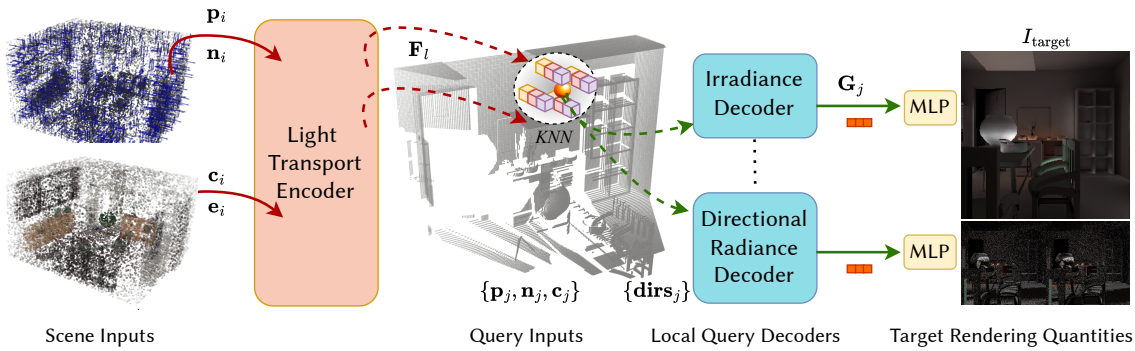


Fig. 4. Our pipeline. Our method first converts the original 3D assets of each scene into a point cloud with associated features (on the left). The scenes are then encoded into a light transport embedding, consisting of latent codes (depicted as small blocks within the dashed circle) anchored at the scene point positions. At render time (center), each query point gathers local latent codes via k-nearest neighbors and feeds them together with the query point attributes into a target-specific query decoder, of which the core is a cross-attention. Note that direct emission can be added on top of the predicted output, so it needs not be a query input.

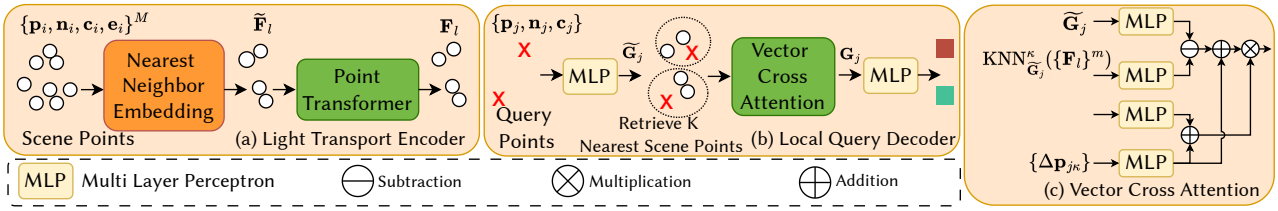


Fig. 5. Network architecture for Light Transport Encoder and Local Query Irradiance Decoder.

As shown in Figure 5(b), this explicitly provides both the local query context ($\widetilde{\mathbf{G}}_j$) and the relative geometry ($\mathbf{P}_{j\kappa}$) needed to weigh the contributions of neighboring scene points.

Vector Cross-Attention. We leverage a transformer to iteratively aggregate features from neighborhood *scene points* onto the *query point*. Crucially, rather than applying the standard dot-product attention between the query and scene points (as query and key, values respectively), we use a vector cross-attention operation. We replace the dot product attention with subtraction as the relation function. Unlike the dot product, which collapses feature dimensions into a single scalar weight, subtraction-based attention computes distinct scores for each channel of the value matrix [Fan et al. 2022; Zhao et al. 2021]. This preserves channel-wise granularity and significantly increases the diversity of feature interactions when aggregating neighborhood information. Additionally, we augment the attention and value matrices with the relative geometry $\mathbf{P}_{j\kappa}$ derived earlier.

$$\mathbf{G}_j^q = \mathrm{W}_q(\widetilde{\mathbf{G}}_j), \mathbf{G}_j^k = \mathrm{W}_k(\mathbf{KV}), \mathbf{G}^v = \mathrm{W}_v(\mathbf{KV}) \tag{8}$$

$$\mathbf{A} = \mathbf{G}_j^q - \mathbf{G}_j^k + \mathbf{P}_{j\kappa} \tag{9}$$

$$\mathbf{G}_j = \mathrm{sum}_\kappa(A(\mathbf{G}^v + \mathbf{P}_{j\kappa})) \tag{10}$$

Formally, as shown in the equations, the learned linear projections W_q ,W_k and W_v project the query ($\widetilde{\mathbf{G}}_j$), key, and value inputs ($\mathbf{KV}$) into their respective latent spaces. The final embedding for each query point, denoted by $\mathbf{G}_j$, is obtained by aggregating the value features using the computed attention values $\mathbf{A}$ through summing up along the κ neighboring points (see Figure 5(b) and (c)).

The final output $\mathbf{I}_{\text{out}}$ is derived as follows:

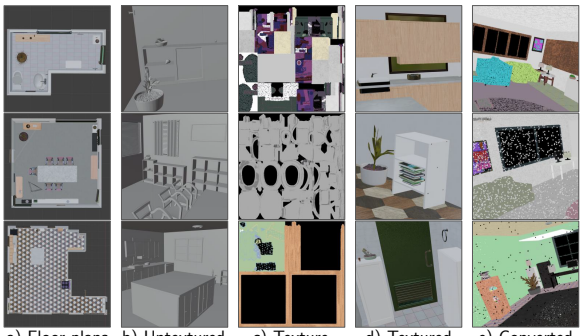


Fig. 6. Dataset overview. Examples of: a) generated floor plans, b) untextured meshes for a floor plan, c) texture atlases, d) textured mesh objects, e) textured meshes converted to scene points directly fed into our pipeline.

$$\mathbf{I}_{\text{out}} = \mathcal{W}_{\text{out}}(\{\mathbf{G}_j^N\}), \tag{11}$$

where $\mathcal{W}_{\text{out}}$ represents the final output projection to convert the set of query points $\{\mathbf{G}_j^N\}$ to the required rendering quantities. These projections are implemented as multi-layer perceptrons.

4 IMPLEMENTATION

Training dataset. We curated a large-scale indoor global illumination dataset comprising 13,900 procedural scenes [Raistrick et al. 2024] spanning diverse floorplans and asset collections (Figure 6). To support future research, we will publicly release all data, code, and trained weights. We generated high-quality ground-truth irradiance by path-tracing approximately 2 million query points per

scene (1,024 rays/point). Additionally, we curated a focused subset featuring glossy materials modeled with the Trowbridge–Reitz (GGX) distribution [Walter et al. 2007] to validate our versatility experiments. Detailed data generation protocols, including geometry refinement and lighting heuristics, are provided in the Supplementary §1.

Training scheme: 3D supervision. To avoid dependence on specific camera views, we train directly on query points uniformly sampled across the entire scene—roughly 2 million points per scene. We train on 90% of the 12k scenes and hold out the rest to test the generalizability. Our model is trained on four A10 GPUs for approximately seven days, using learning rate warm-up followed by cosine decay. Each batch contains 8,192 query points, with a per-GPU batch size of 3. More details can be seen in the Supplementary §1.

Rendering pipeline integration and timing breakdown. We integrate our PyTorch-based network inference with `Falcor`'s framebuffer [Kallweit et al. 2022] via CUDA interop. For the local query decoder, we extend `cudaKDTree` [Wald 2023] to accelerate k-nearest neighbor search. We report average runtime on our test set: (1) The *Light Transport Encoder* takes 208 ms which is dominated by the transformer; however, as our method is view-independent, this cost is incurred only once per scene update and is effectively amortized across frames. (2) The rendering phase *Local Query Decoder* requires 368 ms for generating a 512×512 resolution image on a single NVIDIA RTX 5880 Ada GPU, which includes ∼25–36 ms for GPU KNN search with the rest spent on Python-based attention. Although we have not achieved real-time framerates yet, this suggests that fusing the attention layers into a custom CUDA kernel or applying model distillation could yield significant speedups in future work.

Loss function. We optimize the entire network end-to-end to minimize the relative L2 loss in log space, applied to both estimated irradiance and radiance [Müller et al. 2021]:

$$\mathcal{L} = \frac{1}{N} \sum_j \left(\frac{\log(\hat{\mathbf{I}}_j + 1) - \log(\mathbf{I}_j^{\text{gt}} + 1)}{\log(\hat{\mathbf{I}}_j + 1) + \epsilon} \right)^2 \tag{12}$$

where $\hat{\mathbf{I}}_j$ and $\mathbf{I}_j^{\text{gt}}$ denote the predicted and ground truth values for the j-th query point, respectively, and N represents the total number of query points in the batch.

5 EXPERIMENTS AND ANALYSIS

We primarily validate our framework on Diffuse Global Illumination (§5.2), utilizing smooth irradiance prediction as a benchmark for cross-scene generalization. Conceptually, this serves as a generalizable neural analogue to irradiance caching [Ward and Heckbert 1992], replacing interpolation with robust learned aggregation.

Subsequently, mirroring the evolution toward radiance caching [Jarosz et al. 2008], we demonstrate versatility (§5.3) by re-targeting our encoder to predict spatial-directional radiance fields for glossy materials, achieved by simply exchanging the decoder and applying limited fine-tuning. We conclude with a critical analysis of scalability (§5.4) and detailed ablation studies (§5.5).

5.1 Experimental Setup & Baselines

We compare against representatives from three distinct paradigms to cover the spectrum of neural rendering, thereby complementing recent works [Zeng et al. 2025] with a broader evaluation scope.

Path tracing provides our physically accurate ground truth. We use 64 spp path tracing as a quality baseline, indicative of a real-time sampling budget. *We emphasize that this work does not seek to displace standard path tracing or production-standard PT plus denoising combinations*, but rather to explore whether pure data priors can directly predict global illumination from 3D scene configurations without sampled illumination cues.

Deep shading [Nalbach et al. 2017]. This method predicts GI using screen-space G-buffer attributes and direct lighting. We overfit their model on our test views using ground-truth direct lighting to match their setup. However, unlike our 3D approach, it operates purely in 2D, making it inherently view-dependent and unaware of off-screen geometry. Furthermore, our pipeline learns from raw light sources without requiring rasterized cues.

Hermosilla et al. [2019]. This work is pioneering in 3D light transport learning and close to our setup, but limited to single objects without textures. To adapt it to our dataset, we upgraded the architecture with target-count sampling (for consistent point density), 8× larger feature dimensions, and memory-efficient PointConv [Wu et al. 2019]. Despite this, the model struggled to generalize, requiring per-scene fine-tuning to achieve reasonable results.

RenderFormer [Zeng et al. 2025], a more recent transformer approach. We detail its scalability constraints in §5.4.

5.2 Generalizable Diffuse Global Illumination

We evaluate on our complex indoor dataset (excluding [Zeng et al. 2025] due to OOM on these complex scenes, see §5.4). We present qualitative results on six diverse floor plans in Figure 7, visualizing irradiance without texture modulation to highlight global light transport. Figure 8 shows fully rendered images with texture and direct lighting. In all other cases, visible colors originate purely from indirect lighting.

The hallway and living room setups are especially challenging for path tracing, with the light source placed in a distant corner so parts of the room are occluded from the main illuminated area. Our method consistently outperforms neural baselines and faithfully captures color bleeding from multi-bounce paths and soft shadows.

Minor artifacts remain: light leaks at the toilet base (bath room) and beneath the bowl (dining room 1), and slight color shifts in living room. Quantitative results on the test set are provided in table 1. Extended baseline analysis are provided in Supplementary §3. Furthermore, we demonstrate view consistency and various scene editing applications in Supplementary §4.

Table 1. Quantitative comparison (MSE/SSIM) on 105 test scenes. Note that Deep Shading is overfitted to evaluated views due to poor generalization. Our visual quality significantly outperforms all baselines.

Metric	Deep Shading (overfitting)	Improved [Hermosilla et al.]	Path Tracing	Ours
MSE ↓	0.071	0.171	0.051	0.048
SSIM ↑	0.882	0.775	0.425	0.912

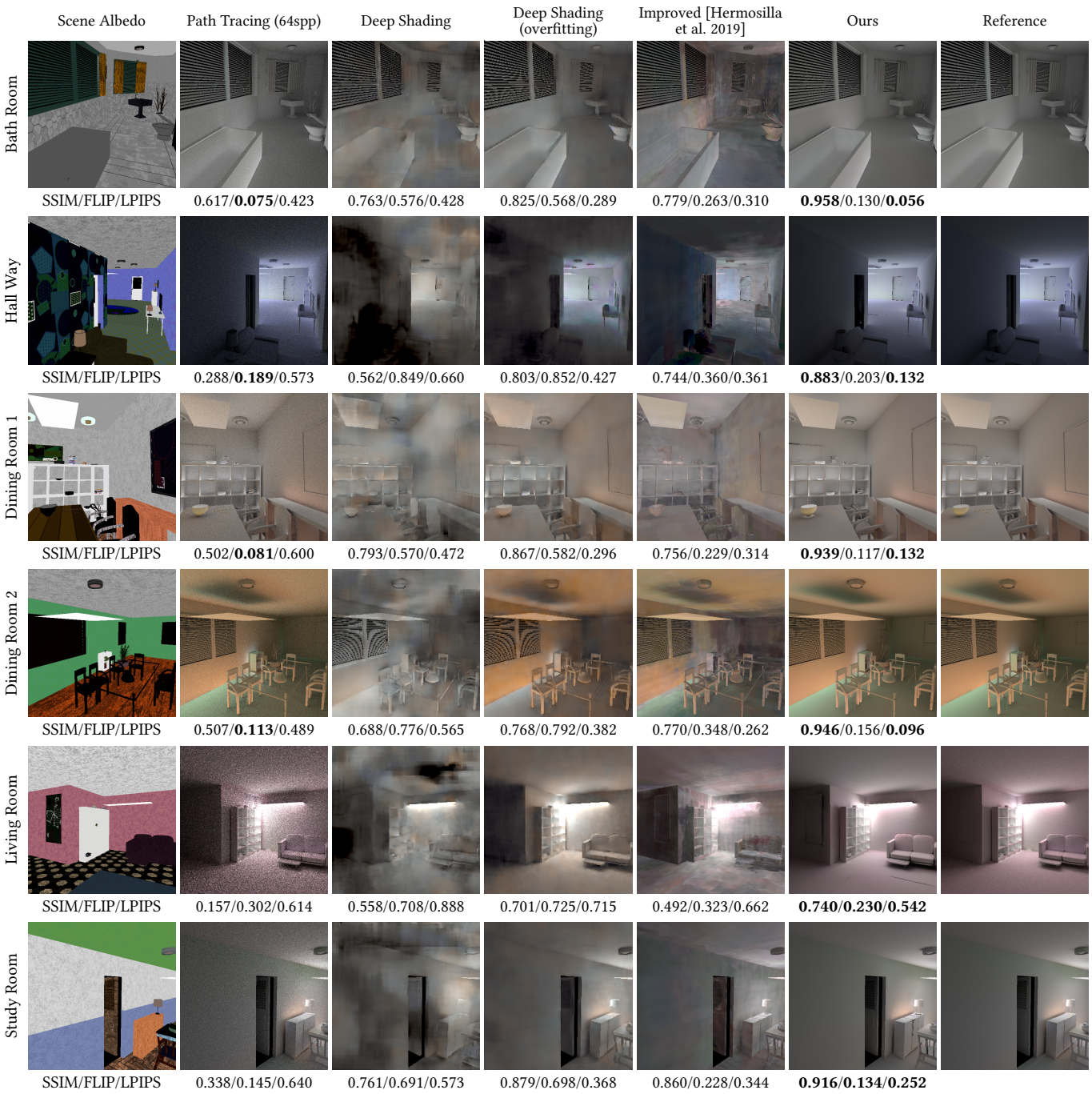


Fig. 7. Irradiance prediction comparison with baseline representatives across six diverse floor-plans under varying illumination conditions. We primarily highlight the substantial qualitative visual improvements while reporting SSIM (↑), FLIP (↓) [Andersson et al. 2020], and LPIPS (↓) [Zhang et al. 2018] as per-scene perceptual quality references. We evaluate our model on a held-out test set to demonstrate its generalization capability. Notably, no per-scene training is performed, unlike prior neural global illumination approaches. For Deep Shading, which fails to generalize to new scenes, we include the test scenes in their training set to show their **overfitting** results here. We extend the original implementation by Hermosilla et al. [2019] to handle our more complex scenes, as their original setup is limited to simple object-level inputs and performs poorly on our dataset (see §5.2). The first column shows the corresponding albedo to provide readers with context for the color of the indirect lighting, while we avoid showing albedo-modulated images to maintain a clear focus on the quality of the predicted irradiance without distraction. We note that path tracing is included as a real-time-budget rendering reference (§5.2) rather than a head-to-head competitor; its error profile is dominated by uniform high-frequency noise, which SSIM and LPIPS reflect more directly while FLIP can score favorably.

Fig. 8. Full renderings with direct illumination and our predicted global illumination for diverse floorplans, lighting conditions and camera views.

5.3 Versatility: Spatial-Directional Radiance Fields

While irradiance evaluates our core capability to encode global interactions and isolates the scalability challenges central to this work, the architecture itself is not tied to diffuse transport. We re-purpose our pre-trained encoder to predict spatial-directional radiance fields, which can be directly integrated with glossy BRDF models, demonstrating versatility to view-dependent light transport.

We construct a focused dataset for these preliminary experiments by moving objects within a living room floorplan and assigning glossy materials using the Trowbridge–Reitz (GGX) microfacet distribution (see Figure 9). We add roughness as a material attribute and use a dedicated Radiance Decoder, taking sampled hemispherical directions to predict directional incident radiance. The ground truth is generated via path tracing using cosine-weighted sampling, discretized into 1024 (32×32) directional bins with 64 samples per bin. We initialize the encoder from our pre-trained irradiance model and fine-tune end-to-end with the new Radiance Decoder, reusing the same encoder backbone for this view-dependent task. This indicates that the learned representation is transferable across rendering tasks rather than tied to its original training objective. Incident radiance field fitting results are visualized in Figure 9.

Our current directional histogram implementation is more expressive than parametric bases such as spherical harmonics for high-frequency glossy structure (see Supp. §4) but remains a discrete, low-resolution representation. Combining the same neural conditioning with compact directional bases such as Spherical Gaussians or Anisotropic Spherical Gaussians, where the network predicts basis parameters rather than per-bin values, is a natural extension we leave as future work. We also explore applications for bootstrapping path guiding in Supplementary §4.

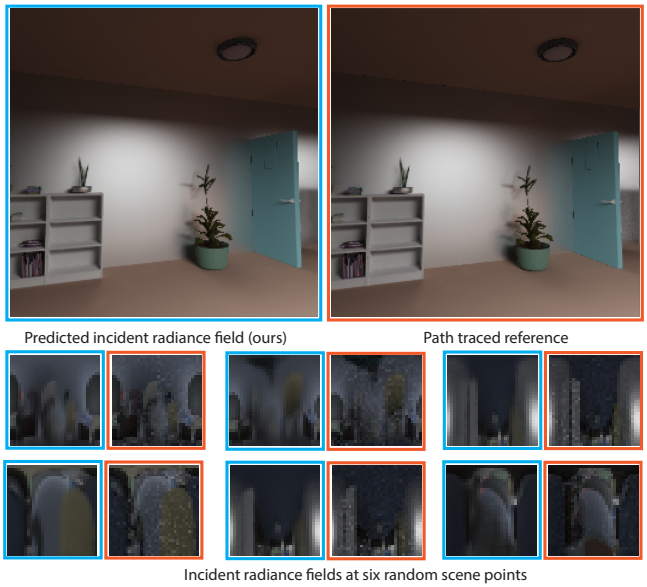


Fig. 9. Rendering comparison for a mixed scene with diffuse and glossy materials. The wall and plant container use a conductor BRDF modeled with a Trowbridge–Reitz (GGX) microfacet distribution and a roughness of 0.1. For glossy surfaces, our Directional Radiance Decoder directly predicts incident radiance given a shading point and any incoming direction. Global illumination (top left) uses BRDF integrated with our predicted directional radiance field, and the top right shows a path traced reference (1024 directional bins per point and 64 samples per bin). The 32×32 slices visualize incident radiance fields at six randomly selected shading points, compared against path-traced reference over 1024 hemispherical directions.

5.4 Scalability and Design Discussion

We analyze the structural divergence between our approach and the vertex-based RenderFormer [Zeng et al. 2025] in Table 2. While both use transformers, these methods occupy distinct points in the design space of the rendering problem and optimize for different goals: theirs focuses on capturing high-frequency transport on object-centric meshes, whereas our design prioritizes the scalability required for general environments, offering support for textures, flexible resolutions and views. As such, the two approaches are complementary rather than competing.

*__Derivation:__ The cost is derived as $N_{\text{patches}} \times \text{Cost}_{\text{patch}}$. With patch size = 1024 and downsampling factor $(1/2)^2$ in our Nearest Neighbor Embedding (§3.3), we have: $\frac{M/4}{1024} \times 1024^2 = 256M$.

Table 2. Structural comparison with RenderFormer [Zeng et al. 2025]. M denotes scene primitives (triangles/points); N denotes query points/ray bundles. Ours achieves linear scalability with significantly fewer parameters. The constant C is explained in the footnote.

Feature	[Zeng et al. 2025]	Ours
1. Model Architecture		
Encoder Complexity	Quadratic $O(M^2)$	Linear $O(CM)^*$, $C = 256$
Decoder Complexity	$O(MN + N^2)$	$O(KN)$, $K = 32$
	Global attention	Local attention
Model Parameters	205M / 483M	42 M
2. Scene Representation		
Input	Mesh Vertices	Sampled Point Cloud
Material Support	Untextured	Textured
3. Training Supervision		
Target	2D Images	Sampled 3D query points
Consistency	View-dependent	View-independent
	Resolution-fixed	Resolution-independent

(1) *The Scalability Barrier.*† RenderFormer employs global attention to model complex long-range interactions. While effective for object-level effects, this has two limitations. First, this incurs quadratic costs across all dimensions: geometry self-attention $O(M^2)$, ray self-attention $O(N^2)$, and cross-attention $O(MN)$. Second, this joint attention mechanism tightly couples scene complexity (M) and rendering resolution (N), creating a "**zero-sum**" competition for the GPU memory. Consequently, its model's learning capacity is constrained by relatively simple training meshes ($\approx 4k$ triangles) and low resolutions. ‡ This bottleneck is illustrated in Figure 2. In contrast, we prioritize linear scalability and decouple the per-query computational cost from the global scene complexity. By leveraging the point representation discussed below, a linear-complexity encoder and a constant local decoder ($K = 32$), we scale to dense indoor scenes with millions of triangles using only **42M** parameters, while supporting arbitrary rendering resolutions.

(2) *Scene Representation.* Vertex-based transformers couple geometry and appearance. In RenderFormer, material attributes are bound to vertices, so capturing high-frequency texture details requires increasing vertex density (tessellation) or encoding attributes at triangle level as in Zeng et al. [2025], which in turn triggers the memory bottleneck. Our point-based approach decouples these factors. We can sample points densely from textured surfaces to faithfully model complex indoor environments with spatially varying materials. This also unifies emissive and non-emissive geometries, making the architecture more adaptable to various light sources. As a quick test to demonstrate that our pipeline can also handle the low-poly regime where RenderFormer natively operates, we additionally fine-tune our model on a small low-poly subset to provide a functional comparison (see Figure 10).

†Recall that M denotes the number of scene primitives (points/triangles) and N denotes the number of queries (query points/pixels/ray bundles). Scalability here mainly refers to computational scalability – the ability of an architecture to handle increasing scene complexity (primitives, resolution, etc.) without a prohibitive increase in resource consumption, specifically GPU memory.

‡We note that the $\approx 4k$-triangle figure refers to the training mesh budget; RenderFormer can be invoked on higher-poly meshes at inference, but at a quality cost reflecting the training distribution.

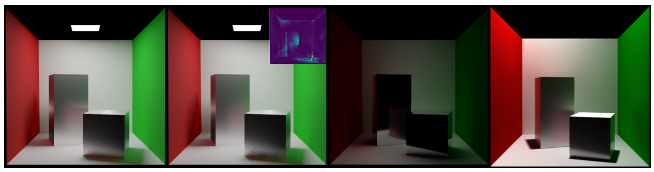


Fig. 10. Cross-method comparison on a Cornell Box, the classical small-mesh setting where RenderFormer is designed to operate. Left to right: path-traced reference; our radiance-decoder prediction (5× error inset [Andersson et al. 2020]); RenderFormer's rendering of the same Cornell Box adapted to its expected input scene assumption, including mesh tessellation, scene normalization, and point-light approximation (see Supplementary §3); and and an official result from the original RenderFormer publication. The latter is included to ensure the comparison remains faithful to their official benchmarks and to provide a reference within the model's intended training distribution. Our model is fine-tuned for one day on a small low-poly subset.

(3) *Supervision.* Training objectives dictate tradeoffs. RenderFormer minimizes loss in 2D image space; its 2D supervision enables it to faithfully model view-dependent caustics and specular highlights, namely, the loss directly sees what the camera sees. However, this approach risks overfitting to constrained viewing frustum and fixed resolution. By supervising directly in 3D world space (§4), we trade off the ability to model high-frequency view-dependent effects for a representation that is inherently view-independent and resolution-agnostic. Both strategies represent valid solutions to distinct optimization problems. See the Supplementary material for further discussion, particularly Supp. Table 4.

5.5 Ablation Studies and Analysis

We validate our architecture through a progressive ablation study, detailed in Figures 11 and 12. The baseline Vanilla model uses a basic global transformer as an encoder. For each query point, it retrieves neighboring scene points via k-nearest neighbors (KNN), followed by simple pooling and a multi-layer perceptron as the decoder. Despite promising results, training suffers from quadratic complexity of global attention, making it inefficient for high-resolution point clouds.

To address this, we substitute the global transformer with our Light Transport Encoder. While this intermediate step (using naïve KNN decoding) incurs a slight performance drop, it establishes the necessary scalability. Our Full Model completes the pipeline by incorporating the Local Query Decoder; this learnable aggregation recovers the lost fidelity and achieves the best overall performance. This ablation study reinforces that each component contributes to the final model's effectiveness. Beyond these core components, a comprehensive suite of additional ablations on further design choices (e.g. scene and query point sampling, hyper-parameters, etc) is provided in Supplementary §2.

6 LIMITATIONS

Our method shares several challenges common to regression-based learning approaches. We observe occasional color shifts in scenes containing underrepresented texture tones, such as the Hallway

and Study room shown in Figure 7. We also note degraded predictions when the primary light source is positioned in an adjacent room (Figure 13). These out-of-distribution shifts could be mitigated through targeted dataset augmentation including outdoor environments, or through limited fine-tuning on specific new domains.

Furthermore, visual artifacts can arise from occluded points. For example, points sampled beneath geometry, such as under a carpet in the 5th scene of Figure 7, can occasionally bleed dark latent codes into visible regions. This issue can be resolved by increasing the neighbor count k during decoding or by pruning fully occluded points during the initial sampling stage.

Consistent with other learning-based global illumination methods, our model may exhibit flickering in dynamic sequences without explicit temporal constraints. It also shows reduced fidelity on high-frequency specular reflections because the current encoder is optimized primarily for diffuse-dominated transport. Finally, while inference latency remains a factor, it can be improved via distillation or embedding pruning. We consider this optimization secondary to our primary goal, which is exploring pure data priors for light transport and architectural scalability.

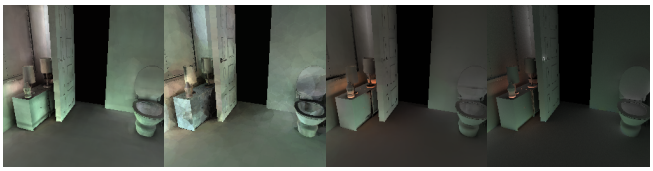


Fig. 11. Qualitative results for ablating different components of the model. This ablation also serves as a **roadmap** of our design and experimental journey. **Left:** Global vanilla Transformer encoder, an intuitive starting point, but quadratic attention makes it non-scalable with point count and hard to train. **Middle:** Our Light Transport Encoder (§3.3) with naive KNN aggregation: highly scalable and fast, but with a noticeable quality drop. **Right:** Adding the Local Query Decoder (§3.4) completes the design and recovers quality, achieving the best overall performance.

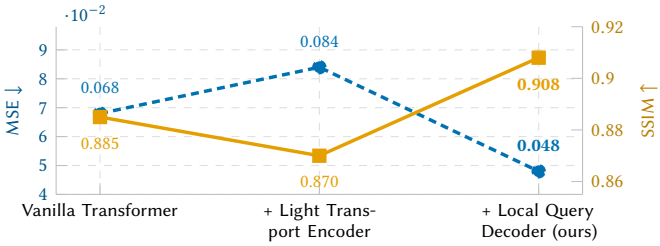


Fig. 12. Quantitative complement (100 random test scenes) aligned with the same ablation roadmap. MSE (left; lower is better) and SSIM (right; higher is better) are reported for the same sequence of design choices.

7 CONCLUSION AND FUTURE WORK

We have presented a scalable 3D light transport embedding that effectively breaks the memory bottleneck of prior neural rendering methods. By replacing quadratic global attention with a linear-complexity transformer on point clouds and employing independent query point-wise local decoding, our framework enables the prediction of global illumination for indoor environments with millions of primitives. Crucially, this efficiency unlocks robust cross-scene generalization. Beyond scalability, we demonstrated the versatility of this generalizable embedding. Our shared encoder successfully adapts from predicting diffuse GI to modeling view-dependent glossy radiance fields. Our 3D supervision ensures view consistency and resolution independence, overcoming the temporal instability often seen in image-space baselines. Our method highlights the potential of tensor cores as an alternative to ray tracing cores, complementing rather than replacing classical pipelines.

Future work includes incorporating participating media and outdoor scenes to extend the spectrum of light transport effects, alongside more sample-efficient training mechanisms to ensure sustainable scaling. Because our pipeline is differentiable by construction, it can serve as a foundational building block for inverse rendering and scene reconstruction. Additionally, incremental embedding updates that re-encode only regions affected by scene changes would extend our amortization beyond camera-only motion to truly dynamic content. Looking ahead, a shared challenge for all efforts in this direction is finding mechanisms that scale sustainably across both the geometric scene space and the high-dimensional rendering parameter space to capture the full complexity of light transport. A careful and evolving balance between data-driven priors and physical laws remains essential for the future of neural rendering.

Fig. 13. Two failure cases discussed in §6: (**left**) color shifting under under-represented texture tones; (**right**) "ajar" lighting where the source is in an adjacent room.

ACKNOWLEDGMENTS

This work was supported in part by NSF grants 2212085, 2105806, 2100237, 2120019, 2127544, and ONR grant N00014-23-1-2526. We also acknowledge support from Apple, gifts from Adobe, Google, Activision, and Qualcomm, the Ronald L. Graham Chair and the UC San Diego Center for Visual Computing. Ramamoorthi acknowledges a part-time appointment at NVIDIA. The authors would like to thank Aaron Lefohn and Chris Wyman for their support; Matt Pharr for valuable input along the way; Thomas Akenine-Möller, Jacob Munkberg, Jon Hasselgren and Zian Wang for their suggestions regarding the evaluation, dataset and training clusters; Miloš Hašan and Benedikt Bitterli for valuable discussions; and to anonymous reviewers for constructive comments. Bing owes particular thanks to Alex Trevithick for pointers to procedural scene generation and many helpful discussions; Zimo Wang and Nithin Raghavan for generously sharing cluster training quotas; Yang Zhou for proofreading and comments; and her fellow interns for helpful discussions in the early stages.

REFERENCES

Pontus Andersson, Jim Nilsson, Tomas Akenine-Möller, Magnus Oskarsson, Kalle Åström, and Mark D. Fairchild. 2020. FLIP: A Difference Evaluator for Alternating Images. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 3, 2 (2020), 15:1–15:23. https://doi.org/10.1145/3406183

James Arvo, Kenneth Torrance, and Brian Smits. 1994. A framework for the analysis of error in global illumination algorithms. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*. 75–84.

Steve Bako, Thijs Vogels, Brian McWilliams, Mark Meyer, Jan Novák, Alex Harvill, Pradeep Sen, Tony DeRose, and Fabrice Rousselle. 2017. Kernel-predicting convolutional networks for denoising Monte Carlo renderings. *ACM Transactions on Graphics* 36, 4 (2017), 97:1–97:14. https://doi.org/10.1145/3072959.3073708

Chakravarty R Chaitanya, Anton S Kaplanyan, Christoph Schied, Marco Salvi, Aaron Lefohn, Derek Nowrouzezahrai, Timo Aila, and Nima Khademi Kalantari. 2017. Interactive reconstruction of Monte Carlo image sequences using a recurrent denoising autoencoder. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 98.

Per H Christensen, Wojciech Jarosz, et al. 2016. The path to path-traced movies. *Foundations and Trends® in Computer Graphics and Vision* 10, 2 (2016), 103–175.

Stavros Diolatzis, Julien Philip, and George Drettakis. 2022. Active Exploration for Neural Global Illumination of Variable Scenes. *ACM Transactions on Graphics* 41, 5 (2022), 171:1–171:18. https://doi.org/10.1145/3522735

SM Ali Eslami, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari S Morcos, Marta Garnelo, Avraham Ruderman, Andrei A Rusu, Ivo Danihelka, Karol Gregor, et al. 2018. Neural scene representation and rendering. *Science* 360, 6394 (2018), 1204–1210.

Zhiwen Fan, Tianlong Chen, Peihao Wang, and Zhangyang Wang. 2022. CADTransformer: Panoptic Symbol Spotting Transformer for CAD Drawings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10986–10996.

Duan Gao, Haoyuan Mu, and Kun Xu. 2023. Neural Global Illumination: Interactive Indirect Illumination Prediction Under Dynamic Area Lights. *IEEE Transactions on Visualization and Computer Graphics* 29, 12 (2023), 5325–5341. https://doi.org/10.1109/TVCG.2022.3209963

Michaël Gharbi, Tzu-Mao Li, Miika Aittala, Jaakko Lehtinen, and Frédo Durand. 2019. Sample-based Monte Carlo denoising using a kernel-splatting network. *ACM Transactions on Graphics (TOG)* 38, 4 (2019), 1–12.

Jonathan Granskog, Fabrice Rousselle, Marios Papas, and Jan Novák. 2020. Compositional neural scene representations for shading inference. *ACM Transactions on Graphics* 39, 4 (2020), 135:1–135:13. https://doi.org/10.1145/3386569.3392475

Miloš Hašan, Fabio Pellacini, and Kavita Bala. 2006. Direct-to-indirect transfer for cinematic relighting. *ACM transactions on graphics (TOG)* 25, 3 (2006), 1089–1097.

Sebastian Herholz, Martin Sik, Lea Reichardt, and Marco Manzi. 2025. Path Guiding in Production and Recent Advancements. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Courses (SIGGRAPH Courses '25)*. Association for Computing Machinery, New York, NY, USA, Article 14, 5 pages. https://doi.org/10.1145/3721241.3733994

Pedro Hermosilla, Sebastian Maisch, Tobias Ritschel, and Timo Ropinski. 2019. Deep-learning the Latent Space of Light Transport. In *Computer Graphics Forum*, Vol. 38. Wiley Online Library, 207–217.

Pedro Hermosilla, Tobias Ritschel, Pere-Pau Vázquez, Àlvar Vinacua, and Timo Ropinski. 2018. Monte carlo convolution for learning on non-uniformly sampled point clouds. *ACM Transactions on Graphics (tog)* 37, 6 (2018), 1–12.

Wojciech Jarosz, Craig Donner, Matthias Zwicker, and Henrik Wann Jensen. 2008. Radiance Caching for Participating Media. *ACM Transactions on Graphics* 27, 1 (March 2008), 7:1–7:11. https://doi.org/10.1145/1330511.1330518

Simon Kallweit, Petrik Clarberg, Craig Kolb, Tom'aš Davidovič, Kai-Hwa Yao, Theresa Foley, Yong He, Lifan Wu, Lucy Chen, Tomas Akenine-Möller, Chris Wyman, Cyril Crassin, and Nir Benty. 2022. The Falcor Rendering Framework. https://github.com/NVIDIAGameWorks/Falcor https://github.com/NVIDIAGameWorks/Falcor.

Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. 2019. PointConv: Deep Convolutional Networks on 3D Point Clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 9621–9630.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics* 41, 4 (2022), 102:1–102:15. https://doi.org/10.1145/3528223.3530127

Thomas Müller, Fabrice Rousselle, Jan Novák, and Alexander Keller. 2021. Real-time neural radiance caching for path tracing. *ACM Transactions on Graphics* 40, 4 (2021), 36:1–36:16. https://doi.org/10.1145/3450626.3459812

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

OpenAI. 2024. Video generation models as world simulators. https://openai.com/index/video-generation-models-as-world-simulators. Accessed: 2025-05-10.

Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. 2017a. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 652–660.

Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. 2017b. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Advances in Neural Information Processing Systems (NeurIPS)*, Vol. 30. 5099–5108.

Nithin Raghavan, Yan Xiao, Kai-En Lin, Tiancheng Sun, Sai Bi, Zexiang Xu, Tzu-Mao Li, and Ravi Ramamoorthi. 2023. Neural Free-Viewpoint Relighting for Glossy Indirect Illumination. In *Computer Graphics Forum*, Vol. 42. Wiley Online Library, e14885.

Gilles Rainer, Adrien Bousseau, Tobias Ritschel, and George Drettakis. 2022. Neural Precomputed Radiance Transfer. *Computer Graphics Forum* 41, 2 (2022), 365–378. https://doi.org/10.1111/cgf.14480

Alexander Raistrick, Lingjie Mei, Karhan Kayan, David Yan, Yiming Zuo, Beining Han, Hongyu Wen, Meenal Parakh, Stamatis Alexandropoulos, Lahav Lipson, et al. 2024. Infinigen indoors: Photorealistic indoor scenes using procedural generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 21783–21794.

Ravi Ramamoorthi et al. 2009. Precomputation-based rendering. *Foundations and Trends® in Computer Graphics and Vision* 3, 4 (2009), 281–369.

Peiran Ren, Jiaping Wang, Minmin Gong, Stephen Lin, Xin Tong, and Baining Guo. 2013. Global illumination with radiance regression functions. *ACM Transactions on Graphics* 32, 4 (2013), 130:1–130:12. https://doi.org/10.1145/2461912.2462009

Paul Sanzenbacher, Lars Mescheder, and Andreas Geiger. 2020. Learning Neural Light Transport. *Arxiv* (2020).

Yawar Siddiqui, Tom Monnier, Filippos Kokkinos, Mahendra Kariya, Yanir Kleiman, Emilien Garreau, Oran Gafni, Natalia Neverova, Andrea Vedaldi, Roman Shapovalov, and David Novotny. 2024. Meta 3D AssetGen: Text-to-Mesh Generation with High-Quality Geometry, Texture, and PBR Materials. In *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (Eds.), Vol. 37. Curran Associates, Inc., 9532–9564. https://proceedings.neurips.cc/paper_files/paper/2024/file/123cfe7d8b7702ac97aaf4468fc05fa5-Paper-Conference.pdf

Peter-Pike Sloan, Jan Kautz, and John Snyder. 2002. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Transactions on Graphics* 21, 3 (2002), 527–536. https://doi.org/10.1145/566654.566612

Cyril Soler, Ronak Molazem, and Kartic Subr. 2022. A Theoretical Analysis of Compactness of the Light Transport Operator. In *ACM SIGGRAPH 2022 Conference Proceedings*. 1–9.

Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Benoit Marcotegui, Francois Goulette, and Leonidas Guibas. 2019. KPConv: Flexible and Deformable Convolution for Point Clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 6411–6420.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

Thomas Vogels, Fabrice Rousselle, Brian McWilliams, Mark Meyer, Steve Bako, Jan Nova´k, Alex Harvill, and Marc Droske. 2018. Denoising with kernel prediction and asymmetric loss functions. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 124.

Jiří Vorba, Johannes Hanika, Sebastian Herholz, Thomas Müller, Jaroslav Křivánek, and Alexander Keller. 2019. Path Guiding in Production. In *ACM SIGGRAPH 2019 Courses* (Los Angeles, California) *(SIGGRAPH '19)*. ACM, New York, NY, USA, Article 18, 77 pages. https://doi.org/10.1145/3305366.3328091

Ingo Wald. 2023. GPU-friendly, Parallel, and (Almost-)In-Place Construction of Left-Balanced k-d Trees. https://arxiv.org/abs/2211.00120

Bruce Walter, Stephen R Marschner, Hongsong Li, and Kenneth E Torrance. 2007. Microfacet Models for Refraction through Rough Surfaces. *Rendering techniques* 2007 (2007), 18th.

Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. 2019. Dynamic Graph CNN for Learning on Point Clouds. *ACM Transactions on Graphics* 38, 5 (2019), 146:1–146:12.

Gregory J. Ward and Paul S. Heckbert. 1992. Irradiance Gradients. In *Proceedings of the Eurographics Workshop on Rendering Techniques*. 85–98.

Wenxuan Wu, Zhongang Qi, and Li Fuxin. 2019. Pointconv: Deep convolutional networks on 3d point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 9621–9630.

Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. 2024. Point Transformer V3: Simpler, Faster, Stronger. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. https://doi.org/10.1109/CVPR52733.2024.00463

Hanggao Xin, Shaokun Zheng, Kun Xu, and Ling-Qi Yan. 2022. Lightweight Bilateral Convolutional Neural Networks for Interactive Single-Bounce Diffuse Indirect Illumination. *IEEE Transactions on Visualization and Computer Graphics* 28, 4 (2022).

Bing Xu, Junfei Zhang, Rui Wang, Kun Xu, Yong-Liang Yang, Chuan Li, and Rui Tang. 2019. Adversarial Monte Carlo denoising with conditioned auxiliary feature modulation. *ACM Transactions on Graphics* 38, 6 (2019), 224:1–224:12. https://doi.org/10.1145/3355089.3356547

Chong Zeng, Yue Dong, Pieter Peers, Hongzhi Wu, and Xin Tong. 2025. RenderFormer: Transformer-based Neural Rendering of Triangle Meshes with Global Illumination. In *ACM SIGGRAPH 2025 Conference Papers*.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *CVPR*.
Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip H.S. Torr, and Vladlen Koltun. 2021. Point Transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 16259–16268.
Chuankun Zheng, Yuchi Huo, Hongxiang Huang, Hongtao Sheng, Junrong Huang, Rui Tang, Hao Zhu, Rui Wang, and Hujun Bao. 2024. Neural Global Illumination via Superposed Deformable Feature Fields. In *SIGGRAPH Asia 2024 Conference Papers*. 1–11.
Chuankun Zheng, Yuchi Huo, Shaohua Mo, Zhihua Zhong, Zhizhen Wu, Wei Hua, Rui Wang, and Hujun Bao. 2023. NeLT: Object-Oriented Neural Light Transfer. *ACM Trans. Graph.* 42, 5 (2023).
Attila T. Áfra. 2024. Intel® Open Image Denoise. https://www.openimagedenoise.org. Accessed: 2025-05-10.

A IMPLEMENTATION DETAILS

A.1 Architecture and training configurations

While the overall architectural design is described in §3 of main paper, we provide the specific layer configurations, channel dimensions and training settings in Table 3.

Table 3. Summary of architectural specifications and training settings.

Category	Parameter	Value
Model	Pre-projection Dim	128
	Nearest-neighbor embedding K	32
	Encoder Backbone	PTV3
	Structure	Enc Depths: [2, 2, 2, 6, 2]
	Channels	[128, 128, 128, 256, 256]
	Patch Size	1024
	Encoder Heads	4 ($D/32$)
	Hidden Dimension (D)	128
	Decoder Layers	6
	Decoder Neighbors (K)	32
	Feed-Forward Dim	512 ($4 \times D$)
	Dropout	0.1
Training	Optimizer	AdamW
	Learning Rate	2×10^{-4}
	Weight Decay	1×10^{-4}
	Batch Size	3
	Total Steps	2,000,000

A.2 Background on Point Transformer V3 and our NNE's receptive field expansion

We adopt Point Transformer V3 (PTV3) [Wu et al. 2024] as our encoder backbone for its scalability properties. While earlier point-based transformers [Zhao et al. 2021] relied on computationally expensive k-nearest-neighbor (KNN) lookups for every attention layer, PTV3 introduces a more efficient serialization-and-patch mechanism.

As illustrated in Figure 14, PTV3 reorders 3D points into a 1D sequence using space-filling curves (e.g., Z-order or Hilbert curves). This sequence is partitioned into non-overlapping patches of a fixed size (we use 1024). Standard self-attention is then computed exclusively within these patches. To ensure cross-neighborhood communication, PTV3 shuffles the serialization order (e.g., swapping between Z-order and Hilbert) across different layers. This design achieves linear complexity, allowing our model to process the millions of points required for high-fidelity light transport.

The spatial coverage limitation. While PTV3 is highly efficient for semantic tasks (like segmentation), its patch-based attention presents a specific challenge for Global Illumination (GI). In GI, a single point's radiance is often determined by distant light sources or large-scale bounces across a room.

A 1024-point patch of raw scene points typically covers only a small, localized 3D volume. Since attention is restricted to the patch, the model can "see" only a thin slice of the environment in a single layer. In standard PTV3, long-range information must propagate slowly through many stacked layers or hierarchical downsampling. This limitation directly motivates our Nearest-Neighbor Embedding (NNE) (§3.3 of the main paper).

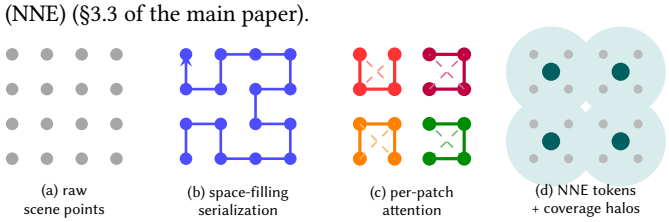


Fig. 14. Visual sketch of PTV3's serialization-and-patch design and our NNE complement. (a) Raw scene points. (b) Points are reordered along a space-filling curve (a Hilbert curve is shown; PTV3 also supports Z-order and transposed variants). (c) The 1D sequence is split into non-overlapping patches (colored); attention is computed only within each patch. (d) Our Nearest-Neighbor Embedding (NNE; §3.3 of the main paper) replaces raw scene points with hierarchically aggregated tokens: two FPS halvings yield $m{=}M/4$ centers (teal), and each token (large teal dot) summarizes a halo (faded teal) of $\sim k^2$ raw points via two stages of k-NN aggregation. A patch of 1024 such tokens covers a much larger 3D region than a 1024-raw-point patch in (c) would.

A.3 Dataset generation details

Our large-scale indoor global illumination dataset is built upon the procedural generation rules from Raistrick et al. [2024]. The dataset comprises 13,900 Blender scenes spanning diverse floorplans and asset collections, including furniture, appliances, and decor.

Geometry refinement. We clean up the geometries to ensure geometric consistency suitable for physically based rendering. This involved: (1) Filtering out low-quality meshes and degenerate geometry. (2) De-duplication: Removing duplicate elements and resolving overlapping exterior shells that could cause light leakage or z-fighting artifacts. (3) All scenes were processed into PBRT format with high-resolution baked textures to preserve material fidelity during export.

Lighting heuristics. We utilized area lights to provide visually pleasing near-field illumination, unifying them with general scene geometries via our point-based representation. To generate realistic configurations automatically, we implemented a heuristic strategy where rectangular lights were instantiated flush with the ceiling to mimic standard fixtures, while enforcing minimum distance constraints from walls and major occluders to prevent unrealistic hotspots or geometric intersections.

Ground truth generation. Each scene is exported to PBRT format, with high-quality triangle meshes and baked high-resolution textures for each asset. The resulting files range from 500 MB to 2 GB.

For ground-truth irradiance computation, we sample 2 million query points per scene and trace 1,024 stratified, cosine-weighted rays per point with a maximum depth of five. Full dataset generation, including mesh processing, texture baking, and rendering, required approximately 250 CPU/GPU days on our compute cluster.

Glossy subset. To support the versatility experiments in §5.3, we generated a focused subset of scenes where glossy surfaces are modeled as conductors using the Trowbridge–Reitz (GGX) microfacet distribution [Walter et al. 2007]. This subset utilizes our standard lighting and geometry pipeline but incorporates additional material roughness attributes and directional incident radiance targets. At each query point, we sampled 1,024 hemispherical directions (32×32) using cosine-weighted, stratified sampling. We traced paths along each direction to compute incident radiance, resulting in a discretized 32×32 directional histogram (and associated PDFs) at every point. We note that this dense representation incurs a significant storage footprint; for future scaling to larger datasets, we recommend adopting on-the-fly BRDF importance sampling to mitigate these memory costs for training data.

Training scene normalization. Light transport is scale-sensitive; phenomena like inverse-square falloff and occlusion depend on absolute physical distances. Therefore, we do not normalize each scene to its own unit cube. Instead, we compute the scene-specific bounding box minimum $\mathbf{b}_{\min}$ and apply a consistent global scale:

$$\hat{\mathbf{p}} = (\mathbf{p} - \mathbf{b}_{\min})/s,$$

where s is a fixed dataset-wide constant representing an upper bound on scene extent. This ensures every training scene fits within $[0, 1]^3$ while maintaining a consistent mapping between world-space and normalized distances. The same transform is applied to query points, while normals and albedos remain unchanged. Under this scheme, the encoder sees physical scale consistently across all environments. Scenes substantially larger than s would either require partitioning or a rescaled s, while the remainder of the pipeline remains unchanged.

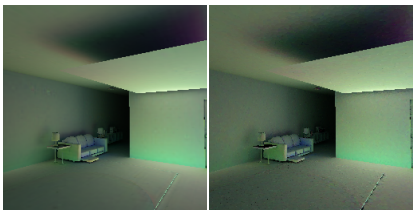
B ABLATION STUDIES AND ANALYSIS

We have made specific design choices regarding training data generation and network architecture. In this section, we present ablation studies to validate these decisions and analyze the impact of key hyperparameters on model performance.

Resource cost and visual artifacts in the design roadmap. We provide further details regarding the ablation roadmap (Fig. 11) by quantifying resource consumption and addressing specific visual artifacts. The Vanilla transformer baseline, while inherently expressive due to its dense attention mechanism, is memory-prohibitive at scale: with $M = 20$k scene points, it exceeds 80 GB of training VRAM at a batch size of 3, necessitating a batch size of 1 in practice. Our Light Transport Encoder reduces peak training VRAM to ≈21 GB at batch size 3, while the full pipeline (including the Local Query Decoder) utilizes ≈22 GB. All configurations were trained for 100k steps.

Notably, the hexagonal patterns visible in the intermediate stage (+Light Transport Encoder) are Voronoi-like artifacts inherent to point-based representations. Since FPS downsampling produces a blue-noise point distribution, naive KNN aggregation implicitly partitions the 3D space into Voronoi cells, resulting in visible discontinuities at cell boundaries. As shown in the final column, our Local Query Decoder successfully resolves these artifacts through learned aggregation, yielding a spatially smooth and high-fidelity output.

Choice of spatial neural primitive. We chose to use point-based neural primitives as the intermediate representation for 3D embedding, where the latent codes are anchored at the scene points. This point-anchor format is naturally suited for the multi-scene encoding via transformers and cross-scene generalization. Several other neural graphics primitives have been proposed for compact scene representation and we acknowledge that there can be potentially better choices. Our preliminary experiments indicate that in single-scene optimization settings, point-anchored latent code displacements may not yet match the convergence efficiency of multi-resolution hash encodings [Müller et al. 2022]. As illustrated in the figure (right), we evaluate this by making our per-point latent codes trainable and utilizing an MLP decoder to regress a single scene. Note that the relative simplicity of the MLP decoders in this experiment may result in lower visual fidelity compared to our primary results; however, they serve as a proof-of-concept for potential architectural extensions.



Integrating our encoder with more expressive neural primitives remains a compelling direction for future work. For instance, adapting the encoder to produce latent features that index into a multi-resolution hash grid [Müller et al. 2022] could potentially combine the benefits of transformer-based scene generalization with the high-resolution spatial encoding and accelerated inference characteristic of grid-based methods.

Scene point sampling strategy. We follow the work by Hermosilla et al. [2018]; Li et al. [2019] to account for the importance sampling densities of points but find their impact on performance negligible. We observe that uniform sampling of 20k points per scene performs comparably to simple heuristics such as biasing toward smaller objects. In general, we find that the effect of sampling strategies diminishes once the scene is sufficiently covered. While performance is not highly sensitive to sampling strategy, the sampling density should still reflect geometric and texture frequencies to capture the signal accurately.

Number of query points for view-independent training. To inform our design choices, we perform a quick verification experiment illustrated in Figure 15. In this single-scene irradiance overfitting setup, we omit the encoder, make the point-based latent codes trainable, and use a simple MLP as the decoder. The experiment evaluates the effect of varying the number of query points. We find that using 2 million query points provides sufficient coverage for complex

Table 4. Ablation results for the number of neighbors (k) during decoding, and number of input scene points. Our choices are colored.

k	PSNR↑	**# scene points**	PSNR↑
8	26.96	5k	28.51
32	34.57	10k	30.94
48	34.71	20k	34.57
		40k	36.25

scenes, offering a good trade-off between performance and memory usage. For simpler scenes, this number can be significantly reduced; for instance, 200k points are sufficient for the Cornell Box.

Neighbor count k and scene-point count M. We further ablate on the number of input scene points and the neighborhood size k used for the KNN search described in §3.4. We conduct these experiments on a small subset of the data, training on 90 scenes and testing on 10 scenes for easy analysis. As shown in Table 4, a small neighborhood ($k = 8$) yields significantly lower performance, with accuracy saturating at $k \geq 32$. Given that larger k values incur substantial computational overhead, we select $k = 32$ as the optimal balance for our final model.

As expected, increasing the number of input scene points M directly improves the performance, and our design choices are made with this in consideration (e.g. efficient light transport embedding §3.3, local decoding §3.4). We observe that beyond $M = 20$k points, PSNR improvements become marginal and visually imperceptible for our benchmark scenes. Consequently, we fix $M = 20$k as our default; however, for environments with higher geometric complexity or larger spatial extent, our architecture is designed to scale with increased point counts accordingly.

Embedding dimension D. The hidden dimension D defines the feature width across our entire architecture. Specifically, the Nearest-Neighbor Embedding projects aggregated tokens into a D-dimensional space; the PTV3 encoder operates within this space across all attention layers to produce the final light transport embeddings $\mathbf{F}_l$; and the Local Query Decoder performs cross-attention at the same dimensionality before final projection. Consequently, D governs per-point representational capacity and maintains a synergistic relationship with M: while M determines spatial coverage, D dictates the richness of the features within that coverage. As shown in Table 5, doubling D approximately doubles training VRAM usage at our current depth and patch size, with a commensurate increase in parameter count. At $D = 256$, training memory requirements (43 GB) exceed the capacity of standard 24 GB GPUs, whereas $D = 64$ provides insufficient per-token capacity for the scene complexity captured by M. We therefore select $D = 128$ as our default, representing the largest dimension feasible on a single 24 GB GPU. For future scaling, D and M should be scaled jointly to accommodate increasing scene complexity.

C EXTENDED BASELINE ANALYSIS

As introduced in §5.1 of the main paper, we compare our method against representatives from three distinct paradigms to cover the full spectrum of neural rendering. This broad evaluation complements recent work Zeng et al. [2025] by assessing performance across diverse architectural philosophies. Below, we elaborate on the experimental setup and discuss the results for each baseline in greater detail.

Table 5. Effect of hidden dimension D on model size and peak training VRAM (batch size 3, M=20k scene points, single GPU). Our default is highlighted.

D	**#Params**	**Train VRAM**
64	10.5 M	10.7 GB
128	42 M	21.3 GB
256	167.7 M	43.1 GB

262k query points | 2M query points | Reference

Fig. 15. We analyze performance under varying numbers of sampled query points on the scene geometry in single-scene overfitting setting, where the model predicts irradiance at primary ray hits.

Deep Shading: screen-space limitations. The deep shading method [Nalbach et al. 2017] is confined to screen-space and has limited awareness of the actual 3D scene. As a result, it struggles to capture effects caused by out-of-sight geometries, exhibits view inconsistencies, and generalizes poorly to unseen scenes. To expose its limitations, we include the test scenes in the training set and show the resulting overfitted predictions in Figure 7. For instance, accurately capturing the shadow cast by a light source onto the ceiling (indirect light bouncing from the floor) in Dining Room 2 is not feasible without access to the 3D knowledge. By using direct lighting as an illumination cue, it was able to predict the area above the light source to be dark, while the region directly beneath it appears brightest (Living Room). We note that the screen-space CNN paradigm is also adopted by Xin et al. [2022]; we use Deep Shading as a representative of this category since both share the same fundamental view-dependence and off-screen-blindness. Recent advances in video foundation models may offer improved spatial reasoning by leveraging the 2D+time dimension. However, with the increasing availability of large-scale 3D data, it may be more effective to bypass the indirect route of reconstructing 3D from 2D projections and instead embrace direct 3D representations—enabling full editability and consistent understanding, as demonstrated by our method.

Hermosilla et al. [2019]: point convolutions on complex scenes. In general, we observe that transformers exhibit strong generalization ability and are surprisingly resistant to overfitting, even on small datasets. In contrast, Hermosilla et al. [2019] is built upon point convolutions, which struggle to learn effectively from our complex dataset. Although we modified their model to better match our

scene complexity, it still fails to capture the intricate nature of global illumination and suffers from noticeable artifacts. We believe transformers offer an advantage due to their ability to model long-range dependencies and dynamically aggregate features across spatially distant regions, which is critical for accurately representing light transport.

Note on path tracing as a baseline. While path tracing often produces noisy outputs, these are typically refined using post-processing denoisers in modern pipelines. We do not claim our method replaces the full path tracing plus denoising setup, but instead emphasize the potential of predicting global illumination directly from 3D scene configurations. Unlike traditional path tracing, where shader execution diverges across pixels due to bounce paths, occlusion, and material interactions, our method performs a uniform forward pass per pixel. This regularity enables efficient parallel execution on modern GPUs and offers predictable performance across varying scene complexity. It also replaces the irregular, memory-bound nature of path tracing with more streamlined and predictable memory access. As power efficiency becomes increasingly critical in modern architectures, transformer-based inference presents a promising direction for scalable light transport.

Discussion on RenderFormer. To further contextualize our contributions, we provide a detailed comparison with RenderFormer, which represents the state-of-the-art in transformer-based global illumination for object-centric tasks. Beyond the scalability differences summarized in Table 2 of the main paper, we expand on the functional capabilities and operational constraints of both methods in Table 6. The two methods diverge fundamentally in their supervision signals and scene representations.

For the qualitative comparison in Figure 10 of the main paper, we took extensive measures to adapt the standard Cornell Box to RenderFormer's specific input assumptions before invoking the released checkpoint. Geometry was tessellated to match their templates. PBRT's diffuse walls map directly to RenderFormer's diffuse term; the PBRT Mirror conductor ($\eta + ik$ with roughness 0.1) is mapped to specular ≈ 0.9 (the Fresnel F_0 at normal incidence) with roughness passed through, plus a small diffuse so that diffuse + specular falls within RenderFormer's training range of $[0.9, 1.0]$. The original area light is replaced by a single emissive triangle at the same centroid, sized to match RenderFormer's training-distribution emitters (similar to point lights) and with radiance set to preserve the original total flux. The flux-correct radiance falls below RenderFormer's training range of $L \in [2500, 5000]\,\mathrm{W/units}^2$, and RenderFormer does not appear to extrapolate linearly outside that range: below 2500 it under-produces light, contributing to the dimmer appearance of its panel. Improving training efficiency by enforcing such physical invariances (e.g., linear radiance scaling) remains an area for future work.

Our model was fine-tuned for one day on a small, low-poly subset to verify its capability in this secondary low-poly regime. While the limited training diversity in this subset results in minor shadow blotchiness, these artifacts are expected to resolve with more comprehensive training on a larger dataset.

Table 6. Capability scope. We demonstrate the capabilities of our method as shown in our results, comparing them with RenderFormer [Zeng et al. 2025] across four key axes: scene complexity, rendering resolution, texture fidelity, and camera freedom.

Feature	Ours vs. RenderFormer
Scene Complexity	RenderFormer is limited to ≈**4k** triangles due to $O(M^2)$ attention costs and the coupling with image resolutions, often requiring decimation that causes geometry loss. In contrast, our method scales via linear attention; our evaluation scenes average **3.2M** primitives (see Table 7).
Resolution	RenderFormer is tied to its fixed training resolution (**512×512**) because queries are coupled by the vision transformer and the resource conflict with geometries. Our method is independent from resolution; Figure 1 demonstrates **2688×1152** resolution with consistent quality.
Textures	RenderFormer assumes constant reflectance per triangle or limited 32×32 textures. Our method supports high-fidelity textures, averaging 40 textures per scene at $\mathbf{1024}^2$ resolution (see Table 7).
Camera Space	We support immersive "walk-throughs" with cameras placed anywhere, while RenderFormer restricts the camera to outside the cornell bounding box.
Material effects	RenderFormer achieves impressive results on high-frequency specular materials by directly predicting outgoing radiance. In contrast, we prioritize view consistency and resolution independence; thus, we predict incident radiance using 3D supervision, which ensures stability across arbitrary camera paths at the cost of some specular sharpness.

Table 7. Example dataset statistics. We show the statistics of our complex indoor scenes exhibiting high geometric density and high-resolution textures, far exceeding the ≈4k primitives limit of Zeng et al. [2025].

Scene Figure	Triangle Count	Texture Resolution
Teaser Figure 1	3,427,263	46×1024^2
Bath Room Figure 7	825,645	29×1024^2
Hall Way Figure 7	2,459,417	41×1024^2
Dining Room 1 Figure 7	2,765,279	35×1024^2
Dining Room 2 Figure 7	4,622,312	44×1024^2
Living Room Figure 7	2,711,230	51×1024^2
Study Room Figure 7	6,044,118	29×1024^2

D ADDITIONAL RESULTS AND APPLICATIONS

D.1 Inherent view consistency

Our model is inherently view-independent (§4 of main paper), as it is trained on query points sampled directly from the 3D scene rather than camera-based observations. Please refer to our supplementary video.

D.2 Scene editing

Light source movement. We assess the model's robustness to changing light positions, as illustrated in fig. 16. In this living room example, moving the area light toward the camera leads to corresponding changes in shadows and indirect illumination. The shadow shifts

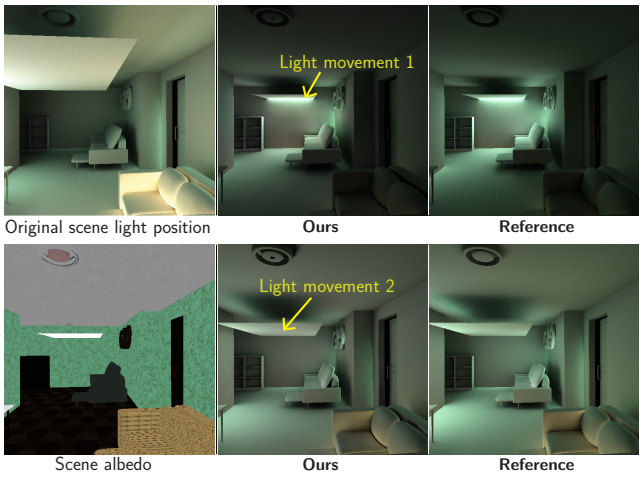


Fig. 16. As the area light moves away or toward the camera, both the shadow and indirect illumination respond accordingly. The model generalizes reasonably well to novel light placements within the same scene. We expect that increasing the diversity of light positions during training would further enhance the quality and robustness of the results. Again, we show the corresponding albedo to provide readers context for the color of the indirect lighting, while we avoid showing albedo-modulated images to maintain a clear focus on the quality of the predicted irradiance without distraction.

relative to the light source due to occlusion of indirect light reflected from the floor, and the two sofa areas become brighter or darker as a result. Although each scene was trained with a fixed light configuration, the model not only generalizes to new floor plans and geometries, but also responds robustly to novel light placements. This behavior reflects the strength of the learned 3D light transport embedding in capturing global illumination dynamics. We expect that training on a wider range of lighting conditions would further enhance both accuracy and robustness.

Object movement. We evaluate the model's robustness to non-emissive object movement, as shown in fig. 17. In this example, as the sofa moves closer to the light source, it becomes more brightly lit, while the small cabinet near the wall appears darker due to increased occlusion. As expected, the shadow consistently follows the moving object.

Material editing. We evaluate the model's response to material editing. In fig. 18, changing the floor's albedo from pale white to deep green causes the predicted irradiance field to update accordingly, modifying the indirect lighting on nearby walls. This result is achieved without any per-scene retraining. Expanding the dataset with a wider range of textures would likely improve robustness in more diverse editing scenarios.

D.3 Architectural versatility and directional bases

While our primary focus is on scalable diffuse transport, the underlying 3D embedding is designed to be task-agnostic. In this section, we expand on the architectural extension introduced in §5.3 of the main paper, focusing on the choice of directional representation for incident radiance.

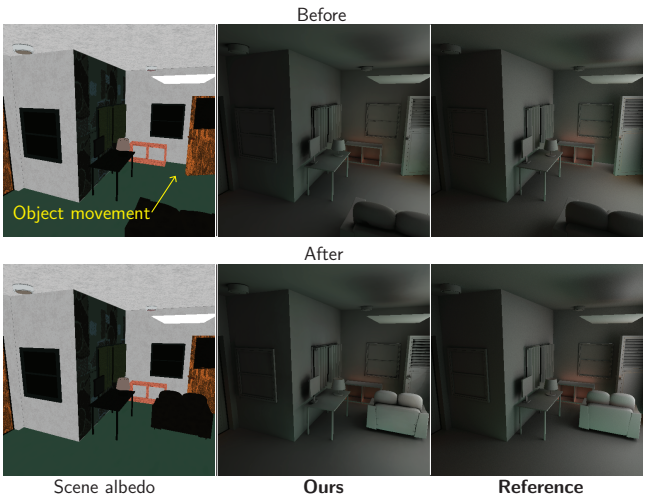


Fig. 17. Irradiance prediction under non-emissive object movement. Unlike previous works, no per-scene training is performed. As the sofa moves toward the light source, it receives stronger indirect illumination, while nearby regions—such as the cabinet near the wall—become darker due to increased occlusion. Again, we show albedo separately to maintain a clear focus on the quality of the predicted irradiance without distraction.

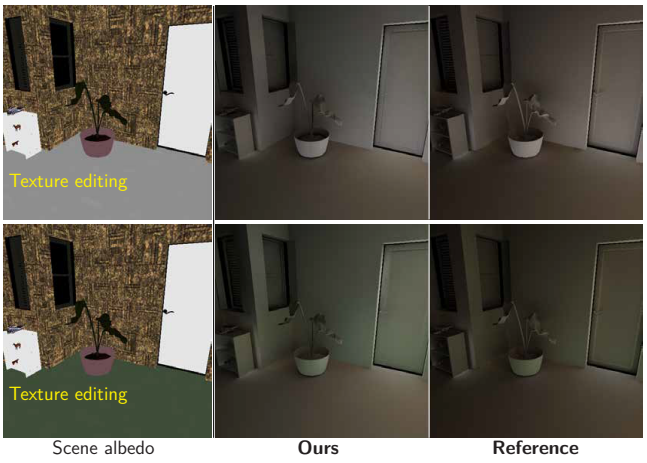


Fig. 18. Irradiance prediction under material editing. When we change the floor's albedo from a pale (whitish) tone to a greenish hue, the indirect lighting cast onto the walls adjusts correctly. No per-scene retraining is needed.

Learned histogram vs. spherical harmonics. We evaluate our neural directional basis against Spherical Harmonics (SH), the standard parametric basis in neural rendering. While SH offers a compact encoding, it faces a fundamental trade-off between smoothness and frequency resolution. Low-degree SH basis functions tend to oversmooth sharp structures, whereas higher degrees ($l_{\max} \geq 4$) significantly increase the feature dimensionality—$(l_{\max}+1)^2$ coefficients per color channel—and often introduce Gibbs-phenomenon ringing artifacts near high-frequency boundaries.

By conditioning on scene geometry and material context, our learned basis achieves significantly higher fidelity for complex incident-radiance distributions. Qualitative comparisons and error

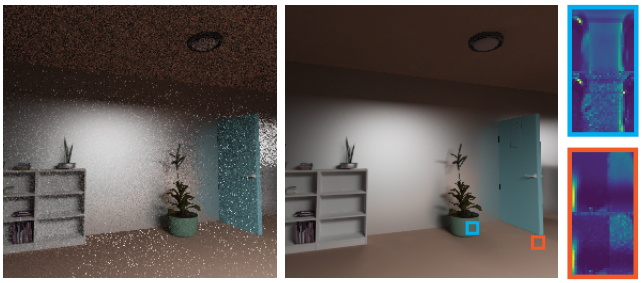


Fig. 19. Path tracing results using different sampling strategies. Left: BRDF sampling with 2 samples per pixel (spp); middle: importance sampling via CDF inversion based on the normalized histogram of the product of our predicted incoming radiance field and brdf values; right: the predicted (top) and path traced (bottom) sampling distributions for the incident radiance integrand at two random points. The reference can be seen in Figure 9 of the main paper. Our model-driven sampling significantly reduces noise in low-sample regimes. As more samples are accumulated, our predicted PDF can serve as an effective initializer for modern path guiding methods, mitigating the cold-start problem inherent in per-scene optimization approaches. Note that the cosine term is included in both of the importance sampling strategies.

vs. degree $l_{\max}$ are reported in fig. 20, where our method preserves sharp detail without ringing and attains lower reconstruction error.

Future extension: parametric neural bases. While the current histogram implementation is more expressive for high-frequency structures, it remains discrete. A natural progression for our framework is to utilize the neural conditioning to predict parameters for continuous bases, such as Spherical Gaussians (SG). This would allow for a more compact representation while maintaining the resolution-independence and view-consistency of our 3D embedding.

D.4 Jump-starting path guiding

While the two previous applications focus on directly predicting global illumination, we provide an initial exploration into how our learned embeddings might support traditional unbiased rendering by aiding importance sampling.

Building on the predicted directional radiance field from §5.3 of the main paper, we explore its use in path guiding initialization. As shown in Figure 19, we compare standard BRDF sampling against importance sampling derived from the product of our predicted radiance field and the BRDF term via CDF inversion. To maintain the unbiased nature of the estimator, the predicted guiding PDF is designed with full support (i.e., non-zero probability for every non-zero bin), ensuring that mathematical correctness is preserved even when using an approximate distribution. At very low sample counts (e.g., 2 samples per pixel), modern path guiding methods ([Vorba et al. 2019], [Herholz et al. 2025]) typically lack sufficient samples to construct a usable distribution. Our learned radiance field serves as a "cold-start" initializer, effectively "jump-starting" the guiding process by providing an informed distribution before scene-specific data is gathered. Our model is not intended as a standalone path-guiding solution, but rather as a prior for established methods that utilize defensive BRDF sampling. This approach draws parallels with meta-learning, where knowledge accumulated across scenes enables fast adaptation to new instances, facilitating efficient light transport simulation without per-scene retraining.

We emphasize that this application is intended as a proof-of-concept; the network currently predicts angularly smooth signals via 32×32 histogram supervision, where each bin represents an average over its respective solid-angle subdomain. Consequently, the result is a low-resolution approximation of the true radiance distribution with an inherent frequency limit. Furthermore, we acknowledge that invoking the full MLP decoder to query directional quantities for each individual sample is currently computationally redundant. This per-query overhead could be bypassed in future work by exploring more efficient pathways to amortize the decoding cost, such as utilizing a CNN-based head to predict a direct bitmap of the radiance distribution or predicting basis coefficients in a single forward pass.

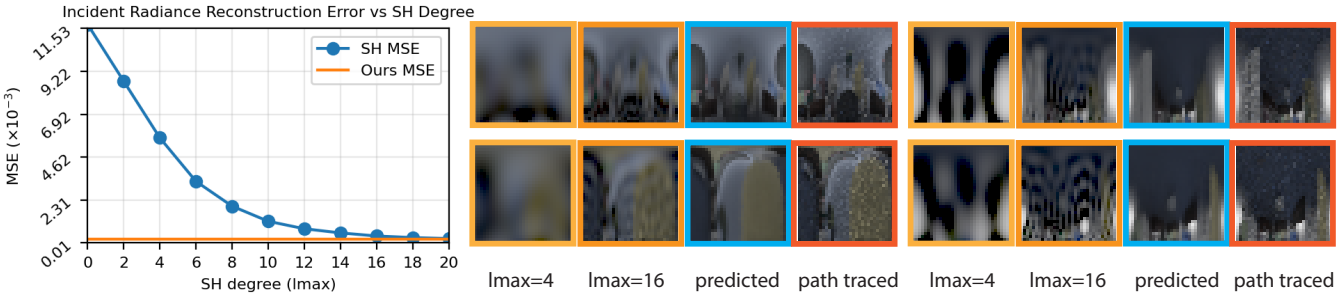


Fig. 20. To assess the effectiveness of our scene encoder and neural basis, we compare our incident-radiance predictions against spherical-harmonics (SH) reconstructions at four randomly sampled surface points. Our method preserves high-frequency glossy reflections. In contrast, low-degree SH yields overly blurred results (first column), while high-degree SH (second column) introduces ringing (Gibbs-like) artifacts. The left plot reports the reconstruction error (MSE) as the SH degree l_{max} increases.