LumiTokens: 3D Relighting via Token-Space Lighting Transformation

Yiwen Chen[1], Matheus Gadelha[2], and Huaizu Jiang[1]

[1] Northeastern University
[2] Adobe Research
https://neu-vi.github.io/LumiTokens

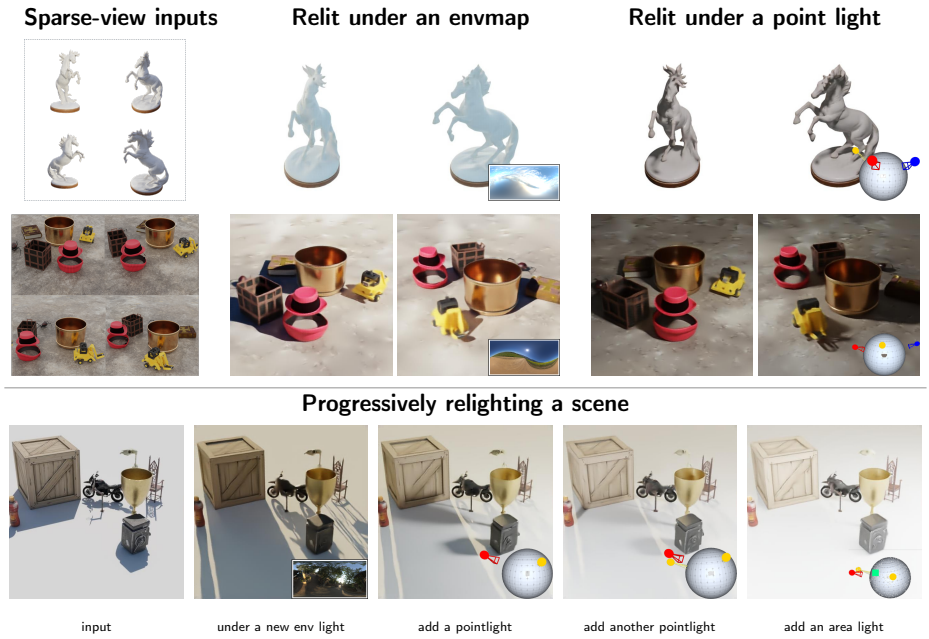

Fig. 1: Given multi-view input images (left), **LumiTokens** relights the scene under diverse lighting types. *Top:* an object relit under an environment map and a point light, rendered from novel views. The red and blue dots indicate the rendering views, yellow dots show the position of the point light sources in 3D, and the cyan rectangle illustrates an area light. *Middle:* relighting generalizes to multi-object scenes with inter-object shadows and reflections. *Bottom:* progressive relighting, where the user incrementally adds light sources to edit the lighting condition.

Abstract. Existing 3D relighting methods operate through either explicit material decomposition, diffusion-based view-space generation, or a combination of both, requiring full recomputation for each new lighting condition. We observe that recent latent scene representations, which

encode multi-view images into a set of compact tokens with no fixed physical semantics, open up a novel design space for relighting. We present **LumiTokens**, a framework that formulates 3D relighting as a direct transformation on latent scene tokens, without explicit 3D representations, rendering equations, or physics-based decomposition. Our model introduces a Scene Token Editor that processes scene tokens jointly with light-ray tokens through self-attention, producing updated tokens that can be decoded into multi-view-consistent relit images. To support diverse lighting types through a unified interface, all lighting signals, including environment maps, point lights, and area lights, are parameterized as Plücker ray tokens, enabling native 3D user interaction with a representation that carries no explicit spatial structure. Crucially, this design supports *progressive relighting*: because the editor's output remains in the same latent space as its input, a user can incrementally build up illumination one light source at a time, with each edit composing in token space. Experiments demonstrate that LumiTokens achieves comparable or superior relighting quality to other methods and supports progressive, composable lighting edits.

Keywords: 3D Relighting · Latent Scene Tokens · Progressive Relighting

1 Introduction

3D scene reconstruction from multi-view captures has seen remarkable progress. Methods such as Neural Radiance Fields (NeRF) [24] and 3D Gaussian Splatting (3DGS) [16] can reconstruct complex objects and scenes with high fidelity, and are increasingly used to create 3D assets for film production, video games, and e-commerce. For these assets to be deployed in new environments, however, they must be rendered under novel illumination: a character scanned in a studio needs to match the on-set lighting of a film, a product captured in a lab must look natural in a customer's living room, and a virtual object in augmented reality should cast shadows consistent with the real world. Relighting, the ability to render a reconstructed scene under arbitrary target illumination, is therefore a critical step for bridging 3D reconstruction and practical downstream applications.

To tackle the relighting problem, existing methods can be grouped into three categories (Fig. 2), each with fundamental tradeoffs. The first relies on **inverse rendering with explicit 3D representations**. These methods recover intrinsic material properties such as surface normals, albedo, and roughness from multi-view images, and relight by re-evaluating a rendering equation under novel illumination [2, 4, 7, 15, 17, 35, 39, 42]. While inherently multi-view consistent due to their explicit 3D structure, these approaches typically require dense multi-view captures and costly per-scene optimization. They are further limited by the expressiveness of their chosen BRDF model and rendering equation.

The second family leverages **diffusion models** as powerful appearance priors for relighting, bypassing explicit material decomposition and producing realistic lighting effects across diverse materials [14, 17, 32, 33, 36]. While early methods

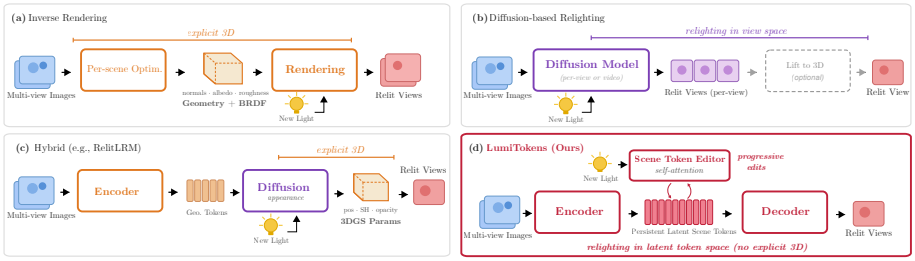

Fig. 2: Comparison of relighting paradigms. (a) Inverse rendering recovers explicit geometry and BRDF properties via per-scene optimization, limited by the expressiveness of the material model. (b) Diffusion-based methods use powerful generative priors to synthesize relit views, without a shared scene representation to anchor cross-view consistency or persist lighting edits. (c) Hybrid methods combine feed-forward geometry encoding with generative appearance synthesis, inheriting limitations from both paradigms. (d) LumiTokens (Ours) performs relighting entirely in latent token space: scene tokens are edited via self-attention with light-ray tokens and decoded into relit views, supporting progressive, incremental lighting edits.

operate on single images, recent extensions adopt video or multi-view diffusion architectures that encourage cross-view coherence through temporal priors or multi-view attention mechanisms [3, 9, 11, 20, 21, 23, 25, 29–31, 41]. Despite their strong results, these methods primarily formulate relighting as image-, video-, or view-space generation: the edited illumination is not stored as a persistent scene-level representation, and each lighting change typically requires a new generative pass rather than an update to a reusable scene state.

A third, **hybrid** category has emerged that draws on both paradigms. These methods build on feed-forward reconstruction models [10, 34], which use transformers trained on massive 3D datasets to predict explicit 3DGS from sparse views in a forward pass. To extend this pipeline to relighting, they incorporate diffusion or generative models that produce lighting-dependent appearance conditioned on a target environment map [19,38]. This combination enables feed-forward relightable reconstruction without per-scene optimization, but the final output remains an explicit 3DGS representation, inheriting the expressiveness ceiling of the 3D parameterization from the first category while still relying on generative appearance synthesis from the second. Across all three categories, relighting is treated as either a decomposition problem, a view-generation problem, or a combination of both. In each case, the lighting edit is not maintained as an incremental update to a persistent scene representation.

Recent advances in novel view synthesis have shown that multi-view images can be compressed into a set of compact tokens that serve as fully learned scene representations, without imposing any explicit 3D structure [1, 12, 13, 27]. Because these tokens encode an entire scene, including geometry and appearance, jointly in a latent space with no fixed physical semantics, we observe that they open up a novel design space for relighting. We propose **relighting as a di-**

rect transformation of a latent scene representation conditioned on lighting. This stands in contrast to all three existing categories: unlike inverse rendering and hybrid methods, our representation is not bound to explicit 3D parameterizations such as 3DGS; unlike prior diffusion-based relighting methods that use generation as the primary mechanism for producing relit views, our relighting operation updates a shared 3D scene representation that naturally anchors cross-view consistency; and unlike all existing paradigms, our approach supports *persistent* editing, where lighting changes accumulate in token space without requiring re-evaluation, regeneration, or re-encoding at each step. To our knowledge, this idea has not been explored in prior work. Realizing it, however, is non-trivial: the latent space is shaped by a reconstruction objective, and it is unclear whether coherent lighting transformations exist as learnable functions in this space. Since geometry and appearance are entangled in the tokens, it is uncertain whether one can change illumination without corrupting the encoded scene structure. And since the tokens carry no built-in 3D structure, it is not obvious how the model can support native 3D user interaction, *e.g.*, responding correctly when a user places a point light at a specific 3D location.

We present **LumiTokens**, a transformer-based framework for relightable novel view synthesis that operates entirely within a learned latent token space, free of explicit 3D representations and rendering equations. Our model adopts a three-part architecture: an *encoder* that compresses a variable number of posed multi-view images (from as few as 4 to as many as 32) into a compact set of 1D scene tokens, a *Scene Token Editor* that transforms these tokens conditioned on target illumination, and a *decoder* that renders the modified scene from novel viewpoints. To support diverse lighting types through a unified interface, we parameterize all lighting signals as Plücker ray tokens: environment maps, point lights, and area lights are all discretized into light rays, enabling native 3D user interaction with a representation that carries no explicit spatial structure. The editor processes scene tokens and light-ray tokens jointly through self-attention, producing updated tokens that the decoder renders into correctly relit novel views. A key design requirement is that the editor's output remains in the same latent space as its input, making the scene tokens a *persistent, editable representation*: a user can incrementally build up the illumination one source at a time, with the editing chain staying in token space and the decoder serving as a read-only preview at any step.

To summarize, our contributions are as follows:

- We demonstrate that 3D relighting can be performed as a direct transformation in a learned latent token space, without explicit 3D representations, rendering equations, or physics-based decomposition.
- We introduce the Scene Token Editor, a transformer module that performs relighting through self-attention between scene tokens and light-ray tokens encoded as Plücker rays, supporting environment maps, point lights, and area lights through a unified interface and enabling native 3D user interaction.

– We show that our formulation supports progressive relighting: because the editor's output remains in the same latent space as its input, lighting edits compose in token space, enabling incremental lighting design that existing paradigms can achieve only through full recomputation at each step.

2 Related Works

Latent Scene Representations. A growing body of work has shown that 3D scenes can be effectively represented as compact sequences of learned tokens, without imposing explicit 3D structure such as NeRF [24], 3DGS [16], or triplanes [5]. SRT [27] introduced this paradigm by encoding multi-view images into set-latent scene representations from which novel views are decoded via cross-attention. LVSM [13] scaled this idea to a large transformer architecture with minimal 3D inductive bias, demonstrating that an encoder-decoder model operating on 1D latent tokens can match or surpass explicit 3DGS-based methods such as GS-LRM [34] on novel view synthesis, particularly on challenging materials involving specular and transparent surfaces. RayZer [12] and its successor E-RayZer [40] further showed that such representations can be learned in a fully self-supervised manner without camera pose annotations. SceneTok [1] demonstrated that unstructured, permutation-invariant tokens can achieve orders-of-magnitude compression of 3D scenes while remaining compatible with generative models. These methods have been used exclusively for novel view synthesis. Extending these representations from passive encoding/decoding to active manipulation for relighting is non-trivial, as discussed in Sec. 3.1. Our work is the first to show that the same class of latent scene tokens can support relighting through a direct, composable transformation in token space.

Inverse Rendering for Relighting. The classical approach to relighting formulates it as an inverse rendering problem: recover intrinsic scene properties (geometry, materials, lighting) from multi-view images, then re-render under novel illumination via a physically based rendering equation [2]. Early neural methods such as NeRD [4], PhySG [35], and NeRFactor [39] extended this paradigm to NeRF-based representations, learning spatially varying BRDF parameters alongside the radiance field. Subsequent work improved efficiency and physical fidelity: TensoIR [15] achieved significant speedups through tensor-factorized representations with online visibility computation, I^2-SDF [42] extended inverse rendering to indoor scenes with near-field lighting, and Relightable 3D Gaussians [7] brought the paradigm to 3DGS with real-time rendering via BVH-based ray tracing. Other methods incorporate learned priors to improve robustness, such as pre-trained depth estimators [18] or diffusion-based material prediction [17, 22]. Despite steady progress, these methods share two fundamental limitations: they are bounded by the expressiveness of their chosen BRDF model, which cannot capture complex effects such as subsurface scattering or inter-reflections, and most require dense multi-view captures with costly per-scene optimization.

Diffusion-based Relighting. An alternative line of work bypasses explicit material decomposition, instead leveraging the strong priors of diffusion models

trained on large-scale data to generate realistic lighting effects. Single-image methods such as DiLightNet [32], Neural Gaffer [14], IC-Light [36], and RGB↔X [33] demonstrated that diffusion models can produce high-quality relightings conditioned on environment maps or lighting descriptors, but each image is relit independently with no multi-view consistency. To address this, recent extensions introduce 3D-awareness through various mechanisms. IllumiNeRF [41] relights individual views with a diffusion model and distills the results into a Latent NeRF via per-scene optimization. DiffusionRenderer [20] uses video diffusion priors with cross-attention injection for temporal consistency. LightSwitch [23] enforces multi-view coherence through multi-view self-attention in the denoising UNet, while UniRelight [9] jointly models albedo estimation and relighting in a single Diffusion Transformer pass. GenLit [3] reformulates relighting as video generation, and ROGR [29] trains a lighting-conditioned NeRF from diffusion-generated relit images. At the scene level, GR3EN [31] and LuxRemix [21] extend generative relighting to room-scale environments with per-luminaire control. Other works explore controllable video diffusion for relighting as part of broader multimodal editing frameworks [11, 25, 30]. Despite impressive quality, these methods lack a shared scene representation to anchor cross-view consistency or persist lighting edits, and each new lighting condition requires a full generative pass.

Hybrid and Feed-forward Relighting. Feed-forward large reconstruction models such as LRM [10] and GS-LRM [34] predict explicit 3D representations (triplane NeRF and 3DGS, respectively) from sparse views in a single forward pass. Several methods extend this backbone to support relighting: RelitLRM [38] adds a diffusion-based appearance generator alongside a deterministic geometry branch, NeAR [19] uses a rectified-flow model to produce lighting-homogenized representations decoded into relightable 3DGS, and GRGS [28] predicts explicit material properties on Gaussians for human relighting. However, their output is an explicit 3DGS rendered through splatting, and each new lighting condition requires regenerating appearance through the diffusion or generative branch. Our approach differs in that relighting operates entirely within a latent token space with no explicit 3D output, and edited tokens persist across lighting changes without regeneration (Sec. 3.1).

3 Method

3.1 Latent Scene Representations for Relighting

Recent work on novel view synthesis has shown that multi-view images can be compressed into compact, *unstructured* token sequences that serve as fully learned scene representations [1, 12, 13, 27]. Unlike explicit 3D representations such as NeRF [24] or 3DGS [16], these methods impose no spatial structure on the latent space: an encoder E maps a set of posed input images $\{I_i, \mathbf{r}_i\}_{i=1}^{N}$ (where $\mathbf{r}_i$ denotes the Plücker ray embedding of the i-th view) into a fixed-size set of 1D scene tokens $\mathbf{z} = E(\{I_i, \mathbf{r}_i\})$, and a decoder D renders a novel view by conditioning on these tokens and a target ray embedding: $\hat{I} = D(\mathbf{z}, \mathbf{r}_t)$.

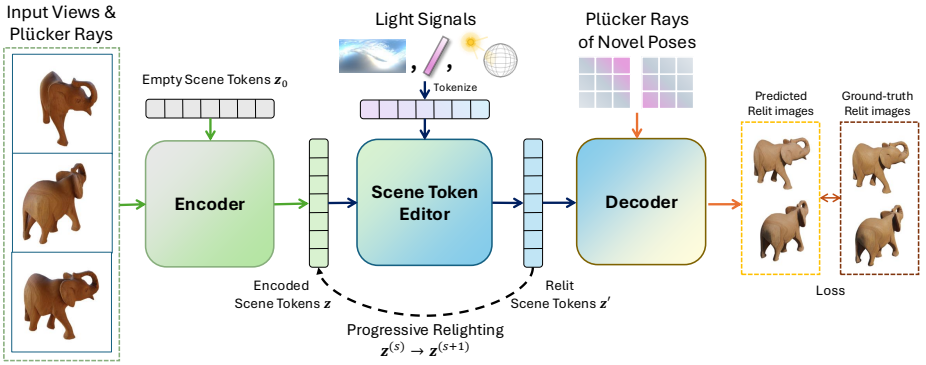

Fig. 3: Overview of the LumiTokens architecture. An encoder maps posed multi-view images, together with learnable empty tokens $\mathbf{z}_0$, into unstructured latent scene tokens $\mathbf{z}$. The scene token editor transforms these tokens conditioned on target lighting: all light types (environment maps, point lights, area lights) are discretized into Plücker ray tokens and processed jointly with the scene tokens through self-attention, producing relit scene tokens $\mathbf{z}'$. A decoder with a DPT head then renders $\mathbf{z}'$ into novel views given target camera Plücker rays. Because the editor's output remains in the same latent space as its input, the architecture naturally supports progressive relighting (dashed arrow): multiple lighting edits chain directly in token space without decoding between steps.

We observe that this paradigm opens up a fundamentally different design space for relighting. Because the tokens encode an entire scene, including geometry and appearance, in a unified representation, relighting can be formulated as *a single transformation on the tokens themselves*, rather than requiring explicit material decomposition, view-space generation, or re-rendering. We propose a novel approach, **LumiTokens**, to exploit this property by introducing a Scene Token Editor T that performs relighting as a direct transformation on the scene tokens: $\mathbf{z}' = T(\mathbf{z}, \boldsymbol{\ell})$, where $\boldsymbol{\ell}$ denotes the lighting signal encoded as a set of light-ray tokens (detailed in Sec. 3.2).

A key design goal is that T approximates a *closed* operation on the latent token space $\mathcal{Z}$: if $\mathbf{z} \in \mathcal{Z}$, then $T(\mathbf{z}, \boldsymbol{\ell})$ should also lie in $\mathcal{Z}$ for any lighting signal $\boldsymbol{\ell}$, as the updated tokens still encode the same scene, just under a different lighting condition. Ideally, the updated tokens $\mathbf{z}'$ are both decodable by D into novel views and editable by further applications of T. This makes the scene tokens a *persistent, editable scene representation*: a user can apply an environment map, then add a point light, then adjust an area light, with each edit accumulating in token space and the decoder invoked only for the final rendering. This stands in contrast to existing paradigms, where each lighting change requires either re-evaluating a rendering equation [7, 15], re-running a full diffusion process [9, 20, 23], or, in hybrid methods such as RelitLRM [38] and NeAR [19], regenerating appearance through a diffusion or generative branch even though the geometry is reusable. To our knowledge, this idea has not been explored in

prior work: existing latent scene representations have been used exclusively for view synthesis [1, 12, 13, 27], and relighting methods have not operated in such a token space.

Realizing this idea, however, poses several challenges. The latent space is shaped by a reconstruction objective, and it is unclear whether coherent lighting transformations exist as learnable functions in this space. Since geometry and appearance are entangled in the tokens, can the editor change illumination without corrupting the encoded scene structure? And since the tokens carry no built-in 3D structure, can the model still support *native 3D user interaction*, *e.g.*, allowing a user to place a point light at a specific 3D location or orient an area light toward a surface, and respond correctly to such spatially grounded editing signals? We address these challenges in Sec. 3.2.

3.2 LumiTokens Architecture

The full pipeline is illustrated in Fig. 3. We adopt the encoder-decoder architecture of LVSM [13] as our backbone and insert the Scene Token Editor T between the encoder and decoder. The editor design operates on generic scene tokens and is not tied to this specific choice.

Encoder. The encoder E is a transformer that maps a sparse set of N posed input images into scene tokens $\mathbf{z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_K\} \in \mathbb{R}^{K \times d}$, where K is the number of scene tokens and d is the token dimension. Each input image is patchified and concatenated with its Plücker ray embedding $\mathbf{r}_i$, forming a set of image tokens. These image tokens are processed jointly with a fixed set of K learnable empty tokens $\mathbf{z}_0$ through self-attention layers; the empty tokens aggregate spatial and photometric information from the input views and become the scene tokens $\mathbf{z} = E(\{I_i, \mathbf{r}_i\}_{i=1}^{N})$.

Light signal tokenization. To address the challenge of interfacing 3D lighting specifications with a spatially unstructured representation (Sec. 3.1), we encode all lighting signals into a unified token format. We discretize the illumination into a collection of light rays sampled from the light source toward the visible scene area, producing a set of L light tokens $\boldsymbol{\ell} = \{\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_L\}$. Each light token is parameterized by its spatial and emissive properties: $\boldsymbol{\ell}_i = (\mathbf{r}_i^{\ell}, \mathbf{c}_i, e_i)$, where $\mathbf{r}_i^{\ell}$ is the ray geometry in Plücker coordinates, $\mathbf{c}_i \in \mathbb{R}^3$ is the spectral intensity (color), and e_i is the radiant power. Following the Plücker ray convention, each light ray is represented as $\mathbf{r}_i^{\ell} = (\mathbf{d}_i, \mathbf{m}_i)$, where $\mathbf{d}_i$ is the unit direction and $\mathbf{m}_i = \mathbf{o}_i \times \mathbf{d}_i$ is the moment vector.

This representation naturally accommodates diverse light source types:

- *Environment maps*: distant illumination with zero parallax with respect to the scene; we nullify the moment vector ($\mathbf{m}_i = \mathbf{0}$), reducing the Plücker representation to a pure directional encoding.
- *Point lights*: rays originate from the source position and are directed toward the scene geometry area, so both the direction $\mathbf{d}_i$ and the moment $\mathbf{m}_i$ encode the light's 3D location.
- *Area lights*: ray origins are sampled across the light's emitting surface, encoding both the position and spatial extent of the source.

This unified tokenization is what enables the native 3D user interaction discussed in Sec. 3.1: regardless of whether the user specifies an environment map, places a point light at a particular 3D location, or orients an area light toward a surface, the lighting signal is converted into the same token format that the editor can process through a single attention mechanism.

Scene Token Editor. The editor T is a transformer that performs a light-conditioned transformation on the scene tokens. Given scene tokens $\mathbf{z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_K\}$ and light tokens $\boldsymbol{\ell} = \{\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_L\}$, the editor concatenates them into a single sequence and processes it through a stack of self-attention layers:

$$\mathbf{z}'_1, \ldots, \mathbf{z}'_K, \boldsymbol{\ell}'_1, \ldots, \boldsymbol{\ell}'_L = \text{Transformer}(\underbrace{\mathbf{z}_1, \ldots, \mathbf{z}_K}_{\text{scene}}, \underbrace{\boldsymbol{\ell}_1, \ldots, \boldsymbol{\ell}_L}_{\text{light}}). \tag{1}$$

The updated light tokens $\boldsymbol{\ell}'_1, \ldots, \boldsymbol{\ell}'_L$ are discarded, and only the updated scene tokens $\mathbf{z}' = \{\mathbf{z}'_1, \ldots, \mathbf{z}'_K\}$ are retained. The use of self-attention, rather than cross-attention, is a deliberate design choice: it allows the light tokens to attend to the scene tokens, enabling the editor to learn where light interacts with scene geometry (*e.g.*, where shadows should fall or highlights should appear), while simultaneously allowing scene tokens to attend to the light tokens to update their appearance accordingly.

Decoder. The decoder D is a ray-conditioned transformer that renders novel views from the scene tokens. Given target camera Plücker ray embeddings $\mathbf{r}_t$, the decoder patchifies the target rays into query tokens and attends them to the scene tokens, producing a set of output tokens that are then mapped to pixels by a prediction head: $\hat{I} = D(\mathbf{z}', \mathbf{r}_t)$. Rather than independently regressing each pixel patch from its corresponding output token with a per-token MLP readout as in LVSM [13], we attach a DPT head [26] to the decoder: it reassembles the decoder's output tokens into multi-scale feature maps and progressively fuses them through convolutional blocks into a dense, full-resolution image. This dense-prediction head recovers noticeably sharper textures and high-frequency details than the per-token MLP readout. Notably, the decoder is applied without modification to both the original scene tokens $\mathbf{z}$ (for novel view synthesis) and the edited tokens $\mathbf{z}'$ (for relighting). The fact that the same decoder produces high-quality images in both cases suggests that the edited tokens remain approximately within the learned latent space, consistent with the design goal described in Sec. 3.1.

3.3 Progressive Relighting

The Scene Token Editor T (Sec. 3.1) is designed so that its output remains in the same latent space as its input, enabling *progressive relighting*: multiple lighting edits composed in token space before a single decoding pass. Formally, given an initial scene representation $\mathbf{z} = E(\{I_i, \mathbf{r}_i\}_{i=1}^N)$ and a sequence of S lighting signals $\boldsymbol{\ell}^{(1)}, \ldots, \boldsymbol{\ell}^{(S)}$, the editing process follows the recurrence

$$\mathbf{z}^{(1)} = T(\mathbf{z}, \boldsymbol{\ell}^{(1)}), \quad \mathbf{z}^{(s)} = T(\mathbf{z}^{(s-1)}, \boldsymbol{\ell}^{(s)}) \quad \text{for } s = 2, \ldots, S, \tag{2}$$

and the final rendering is produced as $\hat{I} = D(\mathbf{z}^{(S)}, \mathbf{r}_t)$. Because each intermediate $\mathbf{z}^{(s)}$ is intended to remain in $\mathcal{Z}$, it can be passed to both the decoder and the next editing step. Each $\boldsymbol{\ell}^{(s)}$ can represent a different light source type: for instance, $\boldsymbol{\ell}^{(1)}$ may specify an environment map, $\boldsymbol{\ell}^{(2)}$ a key light placed at a particular 3D position, and $\boldsymbol{\ell}^{(3)}$ a fill light oriented toward the subject. Fig. 1 shows a qualitative example of this workflow.

This formulation supports *progressive* lighting design: rather than specifying the complete target illumination upfront, the user builds it up one source at a time. The decoder serves as a read-only preview: the next edit always starts from $\mathbf{z}^{(s)}$ itself, not from re-encoded pixels, avoiding the per-step reconstruction loss of a decode-and-re-encode pipeline. To keep the edited tokens within the latent manifold over long chains, we train the editor to predict multiple sequential edits rather than a single transformation (Sec. 3.4), which prevents the distributional drift that a single-step-trained editor would otherwise accumulate. As a result, token-space editing maintains higher fidelity than the pixel-space alternative throughout an interactive lighting session (Sec. 4). To perform incremental editing, existing paradigms have to go through full recomputation at each step: inverse rendering must re-evaluate the rendering equation for the complete lighting configuration, diffusion-based methods must regenerate all views from scratch, and hybrid methods must re-run their appearance branch for each new condition.

3.4 Training Objectives

We optimize the encoder E and the Scene Token Editor T jointly, while keeping the pre-trained decoder D frozen. Fine-tuning E is essential: the token space learned for novel view synthesis must adapt to support lighting transformations, and we find that freezing the encoder degrades relighting quality significantly (see Sec. 4). The total training objective combines a photometric rendering loss $\mathcal{L}_{\text{render}}$ with an invariance regularization $\mathcal{L}_{\text{inv}}$:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{render}} + \alpha\, \mathcal{L}_{\text{inv}}, \tag{3}$$

where α is a hyperparameter balancing the two terms.

Rendering loss. To ensure the edited tokens $\mathbf{z}'$ correctly represent the scene under the target illumination, we supervise the decoded output against a ground-truth relit image I_{gt}. The rendering loss combines a pixel-wise ℓ_2 term with a perceptual term to capture high-frequency details:

$$\mathcal{L}_{\text{render}} = \|I_{\text{gt}} - \hat{I}\|_2^2 + \lambda\, \mathcal{L}_{\text{LPIPS}}(I_{\text{gt}}, \hat{I}), \tag{4}$$

where $\hat{I} = D(\mathbf{z}', \mathbf{r}_t)$ is the image rendered by the decoder at target camera rays $\mathbf{r}_t$, $\mathcal{L}_{\text{LPIPS}}$ denotes the LPIPS perceptual similarity metric [37], and λ controls its relative weight. To support progressive relighting (Sec. 3.3), we do not restrict supervision to a single edit. Instead, we sample a random chain of lighting signals $\boldsymbol{\ell}^{(1)}, \ldots, \boldsymbol{\ell}^{(S)}$, apply the editor recurrently in token space (Eq. 2), and decode

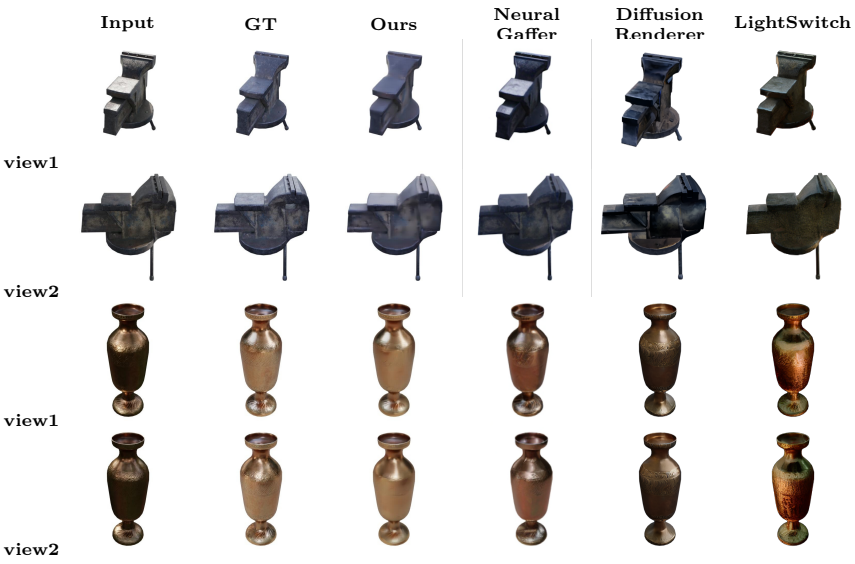

Fig. 4: Qualitative comparison of multi-view relighting. Columns correspond to methods and rows show two viewpoints for each of two scenes. Our method preserves appearance and lighting effects consistently across views, while baselines show artifacts such as residual source illumination and inconsistent highlights.

the intermediate tokens $\mathbf{z}^{(s)}$ to apply the rendering loss at each step against the corresponding ground-truth relit image. Training the editor to predict multiple sequential edits keeps its outputs within the latent manifold over long chains and prevents the distributional drift that a single-step objective would accumulate.

Lighting invariance loss. Since the encoder E receives images captured under a particular source illumination, the scene tokens may entangle scene identity with source lighting effects. To encourage the encoder to focus on the persistent scene content (geometry and materials) and leave all lighting-dependent variation to the editor T, we introduce a lighting invariance loss. Given the same scene captured under two different lighting conditions L_A and L_B, we penalize differences in the encoded tokens:

$$\mathcal{L}_{\text{inv}} = \|E(\{I_i^A, \mathbf{r}_i\}) - E(\{I_i^B, \mathbf{r}_i\})\|_2^2, \tag{5}$$

where $\{I_i^A\}$ and $\{I_i^B\}$ are multi-view images of the same scene under lighting conditions L_A and L_B, respectively. This loss encourages the encoder to factor out the source illumination's contribution, so that the scene tokens $\mathbf{z}$ capture the persistent scene identity (geometry and materials) while leaving the editor responsible for injecting the target lighting. We show in Sec. 4 that this regularization yields a substantial improvement in relighting quality, and that it is as effective as an alternative design that explicitly supervises albedo prediction through an auxiliary decoder head.

Table 1: Multi-view relighting accuracy. ILR: image-level rescaling (per-image scale alignment). SLR: scene-level rescaling (single scale across all views per object).

Method	Image-Level Rescaling (ILR)			Scene-Level Rescaling (SLR)		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
LightSwitch [23]	21.22	0.868	0.105	21.19	0.868	0.131
Neural Gaffer [14]	28.40	0.947	**0.030**	28.23	0.955	**0.037**
DiffusionRenderer [20]	26.39	0.951	0.038	26.24	0.923	0.086
Ours	**30.48**	**0.954**	0.045	**30.36**	**0.917**	0.086

4 Experiments

4.1 Experiment Setup

Data. We train on a large-scale synthetic dataset of 20k objects curated from Objaverse [6], filtered to ensure sufficient geometric and material quality by excluding objects with missing UV maps or non-PBR materials. Each object is rendered in Blender Cycles Engine at 512×512 resolution from 80 viewpoints uniformly sampled on the upper hemisphere. For each object, we generate images under 16 illumination conditions: 4 HDR environment maps sampled from Polyhaven, 4 point light configurations, 2 area light configurations, and 2 combinations of multiple light sources, yielding approximately 20 million images in total. We evaluate on a held-out set of 900 objects (700 from Objaverse and 200 from the Polyhaven model library), with no overlap with the training set. Each test object is rendered under 8 lighting conditions unseen during training.

Evaluation Protocol. We assess two tasks: (1) *Multi-view relighting*, where the model relights all input views at their original poses under a new target illumination, isolating relighting quality from view synthesis; and (2) *Novel-view relighting*, where the model renders relit images from viewpoints unseen during encoding, evaluating joint relighting and view synthesis. For multi-view relighting, the model receives 4 input views. For novel-view relighting, the model receives 8 sparse input views and is evaluated on 8 held-out viewpoints. Following LightSwitch [23], we account for the inherent albedo-exposure ambiguity by applying a least-squares scale alignment to the prediction before computing PSNR, SSIM, and LPIPS [37]. For multi-view relighting, we report metrics under two rescaling protocols: image-level rescaling (ILR), which computes an optimal scale per image, and scene-level rescaling (SLR), which computes a single scale across all views of an object.

Baselines. For multi-view relighting, we compare against diffusion-based methods that can relight individual views: Neural Gaffer [14], DiffusionRenderer [20], and LightSwitch [23]. For novel-view relighting, we compare against inverse rendering methods that perform per-scene optimization: TensoIR [15] and NVDiffrecMC [8]; as well as the multi-view diffusion method LightSwitch [23]. We did not compare against RelitLRM [38], whose code and pretrained models remain unavailable, or NeAR [19], whose implementation was released after our experimental evaluation was completed.

Table 2: Novel-view relighting accuracy. #Input denotes how many source views each method uses; our method requires significantly fewer.

Method	#Input	Synthetic			Objects-with-Lighting			Stanford-ORB		
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
LightSwitch [23]	16	21.61	0.86	0.13	25.43	0.84	0.30	**32.02**	**0.98**	0.03
NVDiffrecMC [8]	50	22.95	0.86	0.10	20.24	0.73	0.40	31.60	0.97	0.04
TensoIR [15]	50	26.17	**0.92**	0.07	26.12	0.77	0.38	25.27	0.94	0.06
Ours	8	**27.76**	0.91	**0.06**	**26.76**	**0.92**	0.36	31.01	0.97	0.03

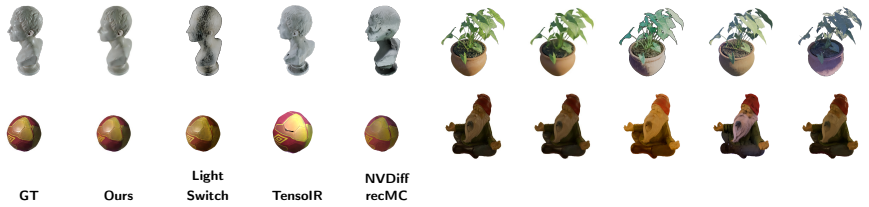

Fig. 5: Visual comparison of novel-view relighting.

Implementation Details. Our backbone follows the encoder-decoder architecture of LVSM [13], with 12 transformer layers each in the encoder, decoder, and Scene Token Editor (token dimension d=768), representing each scene with K=3072 tokens. For the decoder readout, we replace the per-token MLP of LVSM with a DPT head [26] to recover sharper textures and high-frequency details. We provide the full set of training hyperparameters and schedule in the supplementary material.

4.2 Results

Multi-view Relighting. As shown in Tab. 1 and Fig. 4, our method achieves the highest PSNR and SSIM under both rescaling protocols, exceeding Neural Gaffer by 2.1 dB in PSNR and outperforming all baselines in SSIM (0.954). Our SSIM and LPIPS are identical under ILR and SLR, and PSNR drops by only 0.12 dB, indicating consistent relighting across views. On LPIPS, our model (0.045) is slightly outperformed by Neural Gaffer (0.030), the best-performing method on this metric. A possible explanation is that as a diffusion-based model, Neural Gaffer benefits from a generative prior that produces perceptually sharp textures, which LPIPS rewards. Notably, DiffusionRenderer achieves a comparable ILR LPIPS (0.038) but degrades substantially under SLR (0.086), while our LPIPS remains unchanged (0.045 under both protocols), reflecting more consistent perceptual quality across views. Because our model processes all input views jointly through shared scene tokens, it can leverage aggregated multi-view information to separate the object's intrinsic appearance from source lighting effects such as specular highlights and cast shadows. In contrast, single-image base-

lines (NeuralGaffer) relight each view independently, which limits their ability to resolve such ambiguities.

Novel-view Relighting. As shown in Tab. 2 and Fig. 5, our method achieves the highest PSNR (27.76 dB) and best LPIPS (0.069) among synthetic data, and competitive score in realworld bench marks, while requiring only 8 input views, while other baselines require 30-50 views as input. On SSIM, TensoIR achieves the highest score for synthetic data(0.928 vs. ours 0.917). This is consistent with the nature of inverse rendering: by explicitly recovering geometry and materials through dense-view optimization, TensoIR can produce structurally precise reconstructions that SSIM rewards. Our method trades this marginal structural advantage for a large reduction in input requirements and the elimination of per-scene optimization.

4.3 Ablation Studies

Model Design. We ablate two key design choices in Tab. 3: whether to finetune the encoder, and the form of regularization that encourages the encoder to separate scene identity from source lighting. Freezing the encoder, as done in LVSM for novel view synthesis, leads to a 1.1 dB PSNR drop compared to the unfrozen baseline (25.06 vs. 26.12). This confirms that the token space learned for view synthesis alone does not naturally support lighting transformations; the encoder must adapt to produce tokens that are amenable to editing. Adding regularization yields a further improvement of approximately 1 dB. Two forms achieve comparable PSNR: explicit albedo prediction through an auxiliary decoder head (27.09) and our proposed implicit invariance loss (27.05). However, the invariance loss achieves notably better SSIM (0.952 vs. 0.934) and LPIPS (0.045 vs. 0.057), while requiring no ground-truth albedo annotations. We adopt the invariance loss as our default.

Table 3: Ablation study on encoder training and lighting invariance regularization for novel-view relighting.

Encoder	Regularization	PSNR↑	SSIM↑	LPIPS↓
Frozen	None	25.06	0.910	0.073
Unfrozen	None	26.12	0.927	0.046
Unfrozen	Explicit (predict albedo)	**27.09**	0.934	0.057
Unfrozen	Implicit (invariance loss)	27.05	**0.952**	**0.045**

Progressive Relighting. A key property of our framework is that the Scene Token Editor's output remains in the same latent space as its input (Sec. 3.3), enabling multiple lighting edits to be chained in token space without decoding to pixels between steps. To make the editor robust to such chaining, we train it to predict multiple sequential edits rather than a single transformation: at training time we apply a random chain of lighting edits directly in token space and supervise the decoded output at each step. This forces the editor to keep its outputs within the latent manifold over long editing chains, preventing the distributional drift that a single-step-trained editor would otherwise accumulate. We evaluate the model's ability to preserve quality under repeated edits using the following protocol. For each test object, we randomly sample a pool of 5 lighting conditions and render the corresponding ground-truth relit images with the Blender Cycles engine. At each editing step, one lighting

condition is randomly selected from this pool as the new target, and the image rendered under the selected condition serves as the ground truth for that step. We compare two editing strategies over 10 sequential relighting steps: (1) *token-space editing*, where the scene tokens from the previous step are passed directly to the editor together with the new lighting signal; and (2) *pixel-space editing*, where the decoded image from the previous step is re-encoded by the encoder into scene tokens before the editor is applied with the new lighting signal. As shown in Fig. 6, with this multi-step training, token-space editing is consistently better than pixel-space editing across all 10 steps: it preserves fidelity over long chains, whereas pixel-space editing accumulates reconstruction drift from the very first step due to the lossy decode-and-re-encode round trip. This confirms that, throughout the practical range of interactive lighting design, chaining edits directly in token space is the preferred strategy.

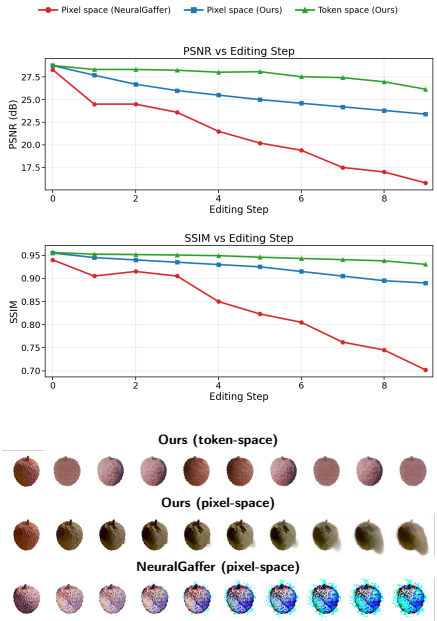

Fig. 6: Progressive relighting ablation. *Top plots:* PSNR and SSIM over multiple editing steps, comparing token-space editing (ours) with pixel-space decode-and-re-encode. Token-space editing preserves fidelity over early steps, while pixel-space editing degrades from the first step onward. *Bottom:* visual comparison across editing steps showing the same trend.

5 Conclusion

We have presented LumiTokens, a framework that formulates 3D relighting as a direct transformation in a learned latent token space. By introducing the Scene Token Editor, which processes unstructured scene tokens jointly with Plücker-parameterized light-ray tokens through self-attention, our method supports environment maps, point lights, and area lights through a unified interface and enables progressive, composable lighting edits entirely in token space. Experiments show that LumiTokens achieves competitive or superior relighting quality compared to inverse rendering and diffusion-based methods that require significantly more input views and costly per-scene optimization. Extending the framework to real-world captures with complex materials and imperfect poses, and training the editor on multi-step chains to improve robustness over longer editing sequences, are promising directions for future work.

References

1. Asim, M., Wewer, C., Lenssen, J.E.: Scenetok: A compressed, diffusable token space for 3d scenes. In: CVPR (2026)
2. Barron, J.T., Malik, J.: Shape, illumination, and reflectance from shading. CoRR **abs/2010.03592** (2020), https://arxiv.org/abs/2010.03592
3. Bharadwaj, S., Feng, H., Becherini, G., Abrevaya, V.F., Black, M.J.: GenLit: Reformulating single image relighting as video generation. In: SIGGRAPH Asia Conference Papers '25. SIGGRAPH Asia Conference Papers '25, Association for Computing Machinery, New York, NY, USA (2025). https://doi.org/10.1145/3757377.3763970, https://doi.org/10.1145/3757377.3763970
4. Boss, M., Braun, R., Jampani, V., Barron, J.T., Liu, C., Lensch, H.P.: NeRD: Neural reflectance decomposition from image collections. In: ICCV (2021)
5. Chan, E.R., Lin, C.Z., Chan, M.A., Nagano, K., Pan, B., Mello, S.D., Gallo, O., Guibas, L., Tremblay, J., Khamis, S., Karras, T., Wetzstein, G.: Efficient geometry-aware 3D generative adversarial networks. In: CVPR (2022)
6. Deitke, M., Schwenk, D., Salvador, J., Weihs, L., Michel, O., VanderBilt, E., Schmidt, L., Ehsani, K., Kembhavi, A., Farhadi, A.: Objaverse: A universe of annotated 3d objects. In: CVPR (2023)
7. Gao, J., Gu, C., Lin, Y., Zhu, H., Cao, X., Zhang, L., Yao, Y.: Relightable 3d gaussian: Real-time point cloud relighting with brdf decomposition and ray tracing. In: ECCV (2024)
8. Hasselgren, J., Hofmann, N., Munkberg, J.: Shape, Light, and Material Decomposition from Images using Monte Carlo Rendering and Denoising. arXiv:2206.03380 (2022)
9. He, K., Liang, R., Munkberg, J., Hasselgren, J., Vijaykumar, N., Keller, A., Fidler, S., Gilitschenski, I., Gojcic, Z., Wang, Z.: UniRelight: Learning joint decomposition and synthesis for video relighting. In: NeurIPS (2025)
10. Hong, Y., Zhang, K., Gu, J., Bi, S., Zhou, Y., Liu, D., Liu, F., Sunkavalli, K., Bui, T., Tan, H.: LRM: large reconstruction model for single image to 3d (2024)
11. Huang, Z., Zhang, M., Wang, R., Tang, R., Zhu, H., Liao, J.: X2Video: Adapting diffusion models for multimodal controllable neural video rendering (2025), https://arxiv.org/abs/2510.08530
12. Jiang, H., Tan, H., Wang, P., Jin, H., Zhao, Y., Bi, S., Zhang, K., Luan, F., Sunkavalli, K., Huang, Q., Pavlakos, G.: Rayzer: A self-supervised large view synthesis model. In: ICCV (2025)
13. Jin, H., Jiang, H., Tan, H., Zhang, K., Bi, S., Zhang, T., Luan, F., Snavely, N., Xu, Z.: Lvsm: A large view synthesis model with minimal 3d inductive bias. In: ICLR (2025)
14. Jin, H., Li, Y., Luan, F., Xiangli, Y., Bi, S., Zhang, K., Xu, Z., Sun, J., Snavely, N.: Neural Gaffer: Relighting any object via diffusion. In: NeurIPS (2024)
15. Jin, H., Liu, I., Xu, P., Zhang, X., Han, S., Bi, S., Zhou, X., Xu, Z., Su, H.: Tensoir: Tensorial inverse rendering. In: CVPR (2023)
16. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Trans. Graph. **42**(4), 139:1–139:14 (2023)
17. Kocsis, P., Sitzmann, V., Nießner, M.: Intrinsic image diffusion for indoor single-view material estimation. In: CVPR (2024)
18. Kuang, Z., Olszewski, K., Chai, M., Huang, Z., Achlioptas, P., Tulyakov, S.: Neroic: Neural rendering of objects from online image collections. ACM Trans. Graph. **41**(4) (jul 2022). https://doi.org/10.1145/3528223.3530177, https://doi.org/10.1145/3528223.3530177

19. Li, H., Ye, C., Chen, H., Xiao, W., Yan, Z., Xiao, L., Chen, Z., Xiang, J., Xu, S., Liu, X., Wang, Y., Zhang, B., Han, X., Yang, J., Zhao, H.: Near: Coupled neural asset-renderer stack. In: CVPR (2026)
20. Liang, R., Gojcic, Z., Ling, H., Munkberg, J., Hasselgren, J., Lin, Z.H., Gao, J., Keller, A., Vijaykumar, N., Fidler, S., Wang, Z.: DiffusionRenderer: Neural inverse and forward rendering with video diffusion models. In: CVPR (June 2025)
21. Liang, R., Müller, N., Weber, E., Zauss, D., Vijaykumar, N., Kontschieder, P., Richardt, C.: LuxRemix: Lighting decomposition and remixing for indoor scenes. In: CVPR (2026)
22. Liang, Z., Zhang, Q., Feng, Y., Shan, Y., Jia, K.: Gs-ir: 3d gaussian splatting for inverse rendering. arXiv preprint arXiv:2311.16473 (2023)
23. Litman, Y., la Torre, F.D., Tulsiani, S.: LightSwitch: Multi-view relighting with material-guided diffusion. In: ICCV (2025)
24. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: ECCV (2020)
25. Poirier-Ginter, Y., Gauthier, A., Philip, J., Lalonde, J.F., Drettakis, G.: A Diffusion Approach to Radiance Field Relighting using Multi-Illumination Synthesis. Comput. Graph. Forum (2024). https://doi.org/10.1111/cgf.15147
26. Ranftl, R., Bochkovskiy, A., Koltun, V.: Vision transformers for dense prediction. In: ICCV (2021)
27. Sajjadi, M.S.M., Meyer, H., Pot, E., Bergmann, U., Greff, K., Radwan, N., Vora, S., Lucic, M., Duckworth, D., Dosovitskiy, A., Uszkoreit, J., Funkhouser, T., Tagliasacchi, A.: Scene Representation Transformer: Geometry-Free Novel View Synthesis Through Set-Latent Scene Representations. In: CVPR (2022)
28. Sun, Y., Wang, C., Zheng, S., Li, Z., Zhang, S., Ji, X.: Generalizable and relightable gaussian splatting for human novel view synthesis. In: SIGGRAPH (2026)
29. Tang, J., Levine, M., Verbin, D., Garbin, S.J., Niessner, M., Martin-Brualla, R., Srinivasan, P.P., Henzler, P.: ROGR: Relightable 3D Objects using Generative Relighting. In: NeurIPS (2025)
30. Xi, D., Wang, J., Liang, Y., Qiu, X., Liu, J., Pan, H., Huo, Y., Wang, R., Huang, H., Zhang, C., Li, X.: CtrlVDiff: Controllable video generation via unified multimodal video diffusion (2025), https://arxiv.org/abs/2511.21129
31. Xing, X., Henzler, P., Hur, J., Li, R., Barron, J.T., Srinivasan, P.P., Verbin, D.: GR3EN: Generative relighting for 3D environments. In: SIGGRAPH (2026)
32. Zeng, C., Dong, Y., Peers, P., Kong, Y., Wu, H., Tong, X.: DiLightNet: Fine-grained lighting control for diffusion-based image generation. In: ACM SIGGRAPH 2024 Conference Papers (2024)
33. Zeng, Z., Deschaintre, V., Georgiev, I., Hold-Geoffroy, Y., Hu, Y., Luan, F., Yan, L.Q., Hašan, M.: RGB↔X: Image decomposition and synthesis using material- and lighting-aware diffusion models. In: ACM SIGGRAPH 2024 Conference Papers. SIGGRAPH '24, Association for Computing Machinery, New York, NY, USA (2024). https://doi.org/10.1145/3641519.3657445, https://doi.org/10.1145/3641519.3657445
34. Zhang, K., Bi, S., Tan, H., Xiangli, Y., Zhao, N., Sunkavalli, K., Xu, Z.: Gs-lrm: Large reconstruction model for 3d gaussian splatting. In: ECCV (2024)
35. Zhang, K., Luan, F., Wang, Q., Bala, K., Snavely, N.: PhySG: Inverse rendering with spherical gaussians for physics-based material editing and relighting. In: CVPR (2021)

36. Zhang, L., Rao, A., Agrawala, M.: Scaling In-the-Wild training for diffusion-based illumination harmonization and editing by imposing consistent light transport. In: International Conference on Learning Representations (2025), https://openreview.net/forum?id=u1cQYxRI1H
37. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR (2018)
38. Zhang, T., Kuang, Z., Jin, H., Xu, Z., Bi, S., Tan, H., Zhang, H., Hu, Y., Hasan, M., Freeman, W.T., Zhang, K., Luan, F.: Relitlrm: Generative relightable radiance for large reconstruction models. In: ICLR (2025)
39. Zhang, X., Srinivasan, P.P., Deng, B., Debevec, P., Freeman, W.T., Barron, J.T.: NeRFactor: Neural factorization of shape and reflectance under an unknown illumination. ACM TOG **40**(6), 1–18 (2021)
40. Zhao, Q., Tan, H., Wang, Q., Bi, S., Zhang, K., Sunkavalli, K., Tulsiani, S., Jiang, H.: E-rayzer: Self-supervised 3d reconstruction as spatial visual pre-training. In: CVPR (2026)
41. Zhao, X., Srinivasan, P.P., Verbin, D., Park, K., Brualla, R.M., Henzler, P.: IllumiNeRF: 3D Relighting Without Inverse Rendering. In: NeurIPS (2024)
42. Zhu, J., Huo, Y., Ye, Q., Luan, F., Li, J., Xi, D., Wang, L., Tang, R., Hua, W., Bao, H., Wang, R.: I^2-SDF: Intrinsic indoor scene reconstruction and editing via raytracing in neural SDFs. In: CVPR (2023)