

Physically Based Rendering in the Latent Space

Vuk Radovanovic¹  Vishesh Gupta¹  Adrien Gruson²  Binh-Son Hua¹ 

¹Trinity College Dublin

²École de Technologie Supérieure (ÉTS)

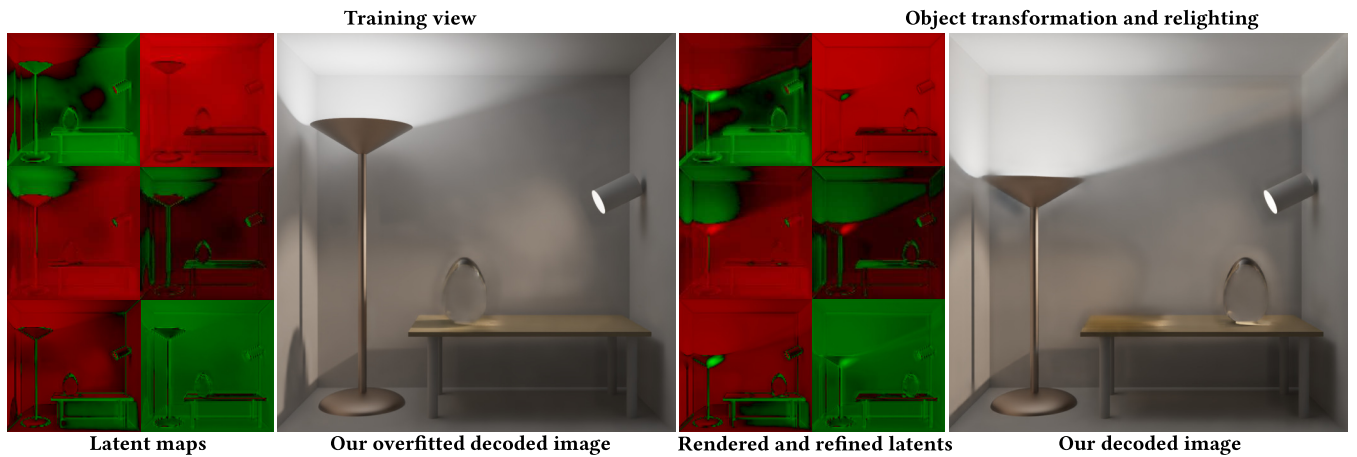


Figure 1: We introduce latent rendering, an image synthesis paradigm that leverages light transport simulation to directly render latent representations of autoencoders. The rendered latents can be subsequently decoded to output final color rendering. This paradigm allows us to perform physically based rendering on 3D scenes to output latents for scene editing applications. We train the renderer using a single image (left), and show its generalized rendering to a scene variant (right, with lamp shortened and glass egg shifted). Latent maps are color-coded, with green for positive and red for negative values, with colors fading to black as values approach zero.

Abstract

Image diffusion models have shown impressive image generation capabilities but are often hard to control, in contrast to classical computer graphics pipelines such as physically based rendering. However, we observe that there is a bridge between light transport phenomena and the distribution of latent space values produced by such models. Thus, we introduce physically based rendering in the feature space learned by the variational autoencoders in generative models, enabling light transport simulation in the latent space. This allows us to leverage physically based rendering techniques to output latent maps for physically guided content generation. We propose modifications to the rendering equation, which, when paired with a differentiable renderer, can yield an optimal set of scene parameters that require only minimal refinement to accurately render into the pretrained latent space. We train our method on a single rendered image, and then demonstrate the generalization of the method to scene geometry changes, lighting changes, and camera view changes.

CCS Concepts

• **Computing methodologies** → **Ray tracing**; **Machine learning**;

1. Introduction

The history of computer graphics is marked by a rich family of techniques aiming to solve the longstanding challenge of realistic image synthesis. Notable approaches include rasterization-based methods for real-time graphics and computer games, along with global illu-

mination methods for high-end animation, movie production, and architectural visualizations. Among these methods, Monte Carlo rendering has been popular across applications because it can simulate physically based light transport effects by numerically solving the rendering integral [Kaj86]. Several rendering algorithms can be

used to estimate this integral, namely path tracing [Kaj86], bidirectional path tracing [VG95,LW93], photon mapping [Jen96,HOJ08], and Metropolis light transport [VG97].

The recent breakthrough of generative modeling in artificial intelligence, i.e., generative AI, has led to radical changes in visual computing, particularly in how visual content is created in computer graphics and computer vision. Particularly, diffusion models [SD-WMG15, SE19, HJA20] demonstrate great promise in solving image synthesis in computer graphics from a new perspective: by generating photorealistic images and videos using natural language and visual descriptions [RBL*22,XCC*25,Bla25,WWA*25]. Such generative models are purely data driven, bypassing the traditional graphics pipeline for visual synthesis.

A key challenge in the latest wave of generative AI models is that these models are not grounded in the laws of physics. Most generative AI models learn features from large-scale datasets and synthesize data by learning and subsequently sampling from the training data distributions. Physical plausibility in visual content generation is mainly driven by data, leaving physical accuracy and correctness largely unconstrained. While computer graphics has numerous techniques targeting physical correctness, generative models lack a principled framework rooted in a physical basis.

This work is our first step toward applying physically based rendering constraints to generative models. We derive a new rendering paradigm that aims to integrate the rigor of light transport simulation with the expressiveness and scalability of visual content generation. We draw inspirations from state-of-the-art image diffusion models, which operate with three main components: a variational autoencoder (VAE), a denoising network, and a prompt conditioning module. Among these, the VAE defines an encoder and decoder that transforms a color image into a latent space that is highly compressed, but remains structured and interpretable; i.e., the latent maps of an image bear high similarity to the geometric and shading structures that appear in the full, uncompressed image. Motivated by this design, we propose *latent rendering*, a governing approach that connects physically based rendering and generative models via the latent space. The goal is to repurpose light transport simulation to operate in this latent space and output latent maps, which can be subsequently reconstructed to color images. By learning to render in the latent space, we aim to understand its connection to traditional RGB rendering, facilitating research toward physically-grounded generative rendering.

Compared to traditional rendering in the RGB color space, latent rendering offers the following benefits. First, latent space is significantly lower-dimensional than RGB space: e.g., a $1024 \times 1024 \times 3$ image could be encoded to a $128 \times 128 \times 16$ latent map, a $12 \times$ reduction that makes rendering operations more compact in the latent space. Second, by rendering into the latent space, we can generalize to render larger latent maps, e.g., $256 \times 256 \times 16$, enabling even higher resolution RGB decoding, e.g., to 2048×2048 . For downstream applications, we demonstrate that it is possible to more efficiently perform language-guided editing on scene representations using score distillation sampling methods [PJB23]. In such applications, latent rendering allows us to circumvent the image encoding step that would otherwise be necessary for an RGB renderer to interface with a latent diffusion model. In our case, gradients

backpropagate to directly update scene parameters without passing through the encoder.

To bring this idea to life, we propose the very first latent rendering algorithm. Our formulation is derived from our analysis of latent representations obtained from encoding a physically based rendering. This analysis motivates us to adapt light transport simulation to reproduce the visual effects perceivable in the latent space, leading to our latent rendering formula. This latent rendering process can be optimized based on differentiable rendering and refined using a neural network, making the rendered latents accurately match ground truth representations using only a single RGB rendering for training. The latent representations can then be edited via 3D scene variants under camera movement, light source changes, or object motion, to produce updated latents that can be reconstructed to color rendering. To summarize, our main contributions are:

- A thorough analysis of the latent space of a pretrained autoencoder through the lens of physically based light transport;
- A latent rendering formulation including a modified rendering equation and a neural refiner which depart from its original physical basis, but provide adequate expressibility to mimic generative latent representations;
- A pipeline allowing to freely edit camera poses, light configuration, and object position directly in latent space for physically based scene-aware image editing.

Our code is available at <https://github.com/trinity-graphics/latent-rendering>.

2. Related Work

2.1. Forward and inverse physically-based rendering

In computer graphics, light transport simulation has been the core of physically based rendering over the past decade [PJH23]. The rendering equation [Kaj86] is the central formulation that dictates how light emitted from sources interacts with 3D objects through their associated bidirectional scattering distribution functions (BSDFs) and is ultimately captured by a virtual camera to produce a photorealistic image. Different solutions to this rendering equation exists, such as path tracing [Kaj86], photon mapping [Jen96, HOJ08], and Markov Chain Monte Carlo rendering [VG97]. Among these, path tracing [Kaj86] remains the de facto standard used in industry, with several improvements such as path guiding [HSRM25], path reuse [BSH02], and reservoir sampling [WKL*23]. A key benefit of physically-based rendering is that this rendering pipeline is fully explainable and controllable, offering fine-grained and precise control of image appearance through explicit scene editing using interactive authoring tools [Ble25].

Light transport simulation can be repurposed to operate in spaces other than the original RGB space. Gradient-domain rendering [LKL*13,KMA*15,BPE17,HGP*19] simulates light transport to estimate image gradients, which are used subsequently for RGB reconstruction [YLWX25], taking advantage of the spatial smoothness of rendered images. In gradient-domain rendering, these gradients remain linear to their underlying pixel values. Our latent rendering instead operates in a non-linear compressed data representation learned by an encoder-decoder pair.

Light transport simulation could also be made differentiable for inverse rendering tasks [LADL18, NDVZJ19, ZMY*20, ZCZ25], e.g., for optimizing scene parameters to match a target image through gradient backpropagation. By applying Reynolds’ transport theorem [Rey03], the differentiation of the rendering equation yields an interior and an exterior term [ZCZ25]. The exterior term models signal discontinuities such as visibility changes, and must be handled when optimizing scene geometry, typically requiring specialized techniques [ZRJ23, XBLZ24, SPM25]. The interior term describes how scene parameters, such as BSDFs or emitter profiles, influence the rendered image. These interior derivatives can be efficiently evaluated using backward radiance formulations [NDSRJ20], e.g., path-replay backpropagation [VSJ21]. In this work, we built on top of differentiable rendering to redirect light transport simulation to operate in the latent space, exclusively focusing on the interior component for optimizing emitter and BSDF parameters.

2.2. Generative models

In contrast to physically based rendering, generative models offer a new paradigm for synthesizing photorealistic images by learning from data, e.g., using diffusion models [SDWVG15, HJA20, SE19]. Latent diffusion models [RBL*22, XCC*25] perform the diffusion process in highly compressed but interpretable latent spaces, enabling large-scale training and high-resolution image generation. Video diffusion models [YTZ*25, WWA*25] are recent advances that extend image diffusion models for generating animated scenes. While the latent spaces of these generative models are highly expressive, controlling the outputs of diffusion models is a challenging task. General methods such as low-rank adaptation [HSW*22] and ControlNet [ZRA23] adjust a subset of parameters of the diffusion models to induce specific visual styles, with optional extra inputs such as depth maps or user-provided strokes to control the generation process. The denoising diffusion process could also be intercepted with visual, semantic, or user guidance at early timesteps to provide high-level control for image editing [MHS*22]. Alternatively, specific solutions are required for particular tasks such as intrinsic decompositions [LDHG*25, HWH*25], material generation [KHN25], relighting [ZDG*24, BFB*25], or novel view synthesis [LWVH*23, ZGV*25]. However, these approaches do not offer sufficiently precise or consistent control over camera parameters, lighting conditions, and object placement simultaneously.

2.3. Hybrid rendering

There exist ongoing efforts in combining generative models with traditional rendering concepts to model physically based effects and improve controls over image outputs. For example, transformer-based models have been proposed to directly predict approximate solutions to the rendering equation [ZDP*25], but they remain limited to relatively small scenes. Video diffusion models can be tuned to output material and environment maps, enabling the generation of material and lighting effects in videos without light transport simulation [LGL*25]. Neural renderers could be used to refine physically based rendering results, enabling tasks such as scene relighting from a single image [CA25] when combined with monocular geometry estimation [WXD*25].

A few previous works exploit the latent space for feature rendering. Latent-NeRF [MRP*23] performs text-to-3D generation by directing a neural radiance field to output latent features compatible to a pretrained diffusion model, as guided by a score distillation objective. Splatent [HSHS*26] reconstructs 3D Gaussian Splats in the latent space of a VAE, and leverages a one-step diffusion process to improve the quality of the latents, thereby enhancing novel view rendering. Our work shares the high-level idea of rendering to the latent space as we aim to use the latent space to establish a synergy between latent diffusion models and physically based light transport simulation. As our method is built on physically based rendering, relighting is naturally supported and scene editing is generally more intuitive and controllable.

3. Background

3.1. Physically based rendering

We are interested in solving the light transport equation [Vea97], which, a light path $\bar{\mathbf{x}} = \mathbf{x}_0, \dots, \mathbf{x}_k$ consisting of k vertices, contributes to pixel i by evaluating the following integral over the path space $\mathcal{P}$:

$$I_i(\pi) = \int_{\mathcal{P}} f_i(\bar{\mathbf{x}}, \pi) d\bar{\mathbf{x}} = \int_{\mathcal{P}} r_i(\bar{\mathbf{x}}) T(\bar{\mathbf{x}}, \pi) L_e(\mathbf{x}_{k-1} \leftarrow \mathbf{x}_k, \pi) d\bar{\mathbf{x}}. \quad (1)$$

Here, $r_i(\bar{\mathbf{x}})$ denotes the pixel filter, $T(\bar{\mathbf{x}}, \pi)$ the path throughput, and $L_e(\mathbf{x}_{k-1} \leftarrow \mathbf{x}_k, \pi)$ the emitted radiance at the light source. The throughput is defined as $T(\bar{\mathbf{x}}) = \prod_{j=1}^{k-1} f_r(\mathbf{x}_j, \pi) \cos \theta_j$, where f_r is the bidirectional scattering distribution function (BSDF) evaluated at each surface interaction, and θ_j is the angle between the incident direction at vertex j and the surface normal. The integral depends on scene parameters π , such as BSDF parameters and emitter intensity values. The integral in Eq. (1) can be solved using Monte Carlo estimation. By default, the integral is evaluated in standard color spaces such as the RGB and produces non-negative estimates that respects physical constraints imposed by light transport.

3.2. Differentiable rendering

Physically based rendering can be used in an optimization framework to determine optimal scene parameters $\hat{\pi}$ so that the rendering matches a target image I_{gt} :

$$\hat{\pi} = \arg \min_{\pi} \mathcal{L}(I(\pi), I_{\text{gt}}), \quad (2)$$

where $\mathcal{L}$ is a differentiable objective function such as the mean squared error (MSE). Gradient-based optimizers, such as Adam [KB15], can be used to solve this optimization problem, but they require access to how changes in the scene parameters affect the rendered image. By the chain rule, this requires computing the gradient of each pixel intensity with respect to the scene parameters:

$$\partial_{\pi} I_i(\pi) = \partial_{\pi} \int_{\mathcal{P}} f_i(\bar{\mathbf{x}}, \pi) d\bar{\mathbf{x}} = \int_{\mathcal{P}} \partial_{\pi} f_i(\bar{\mathbf{x}}, \pi) d\bar{\mathbf{x}}. \quad (3)$$

Transferring the derivative inside the integral requires care when the scene parameters control discontinuities in the rendering process [Rey03]. Several techniques for differentiable Monte Carlo rendering exists [LADL18, VSJ21], but in this work, we assume that our scene parameters vary continuously over the integration

domain, and their derivatives could be efficiently estimated using path replay backpropagation [VJS21].

3.3. Latent spaces

Modern generative models [RBL*22, Bla25] are typically trained in two stages. First, an autoencoder is trained to learn a latent space that encodes data into latent variables and reconstructs the original data from these latent representations. Second, a denoising network is learned to map random noise to latent codes. In this work, we focus on the first stage, namely the autoencoder and its latent representations.

An autoencoder is composed of an encoder $\mathcal{E}$ and decoder $\mathcal{D}$. Given an image $x \in \mathbb{R}^{H \times W \times 3}$, the encoder produces a compressed latent representation $z = \mathcal{E}(x) \in \mathbb{R}^{h \times w \times d}$ such that $h < H$, $w < W$, and $d > 3$ where H, W, h, w are image and latent height and width, and d is the number of latent channels, respectively. We can then decode this latent representation to reconstruct an image $\tilde{x} = \mathcal{D}(z) \in \mathbb{R}^{H \times W \times 3}$ similar to original image x . This autoencoder is trained to minimize the difference between the distribution of x and $\tilde{x}$, following the principles of variational autoencoders (VAEs) [KW14]. From empirical designs of image diffusion models, typically we have $h = H/8$, $w = W/8$, $d = 4$ for Stable Diffusion 2.1 and SDXL [RBL*22], and $d = 16$ for Stable Diffusion 3.5 and FLUX [EKB*24, Bla25], resulting in latent compression ratio of 48 and 12, respectively.

4. Latent Space Analysis

This work aims to align rendering values produced by a physically based renderer with autoencoder latents, enabling the renderer to generate latent representations that can subsequently be decoded into color images. Let us first analyze these latents to identify the requirements for adapting a physically based renderer to the latent space. Here we focus on the latent space defined by the pretrained VAE in Stable Diffusion 3.5 [EKB*24], an open-weights text-to-image diffusion model for high-quality image generation. We use image resolution of 1024×1024 , latent resolution 128×128 with $d = 16$. We analyze the latents of the classic Cornell box as shown in Fig. 2. We observe the following phenomena that makes rendering in this latent space inherently different from RGB rendering:

1. In the latent space, emitted light is signed (Fig. 2, A), in contrast to the strictly non-negative values assumed in physically based rendering.
2. The latents encode scene structure with a strong emphasis on geometric edges and corners (Fig. 2, B). In regions with extremely high-frequency content, latent values may become perceptually indistinguishable from noise.
3. In occluded regions, latent values can exhibit both positive and negative energy, which can be an even higher intensity than surrounding unoccluded regions (Fig. 2, C). This differs from physically based rendering, where occlusion from a light source can only reduce the amount of energy received by a surface.
1. There is structural parity between the RGB image and the corresponding latent channels, in which different BSDFs and objects receive distinct latent values that mirror their RGB counterparts.

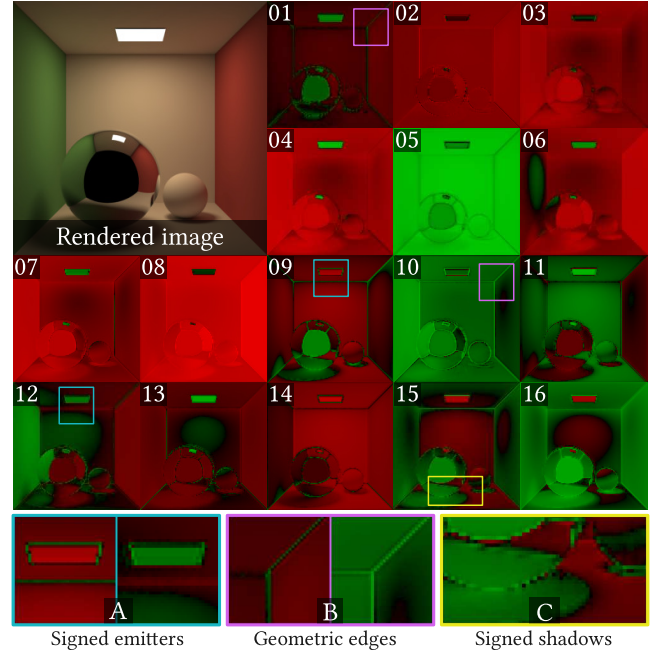


Figure 2: Latent maps of the Cornell Box rendering. The zoom-in patches demonstrate visual effects unique to latent representations, including (A) the signed emitter, (B) highlighted geometry edges, and (C) signed shadows, which cannot be rendered using traditional physically based rendering.

2. Signed energy emitted in the latent space is reflected throughout the scene and is closely preserved on perfectly specular surfaces.
3. As the incident angle of light varies smoothly, the intensity of the reflected signed energy varies accordingly. This effect is more pronounced in some channels and less so in others, with certain channels exhibiting a flatter, more uniform appearance (e.g., Fig. 2, channel 02).

In the literature, there exist some attempts in modeling the relationship between latents and RGB values directly, such as a 4×4 linear mapping in Stable Diffusion 1.4 [Tur22]. However, such mapping is applied for general images and to each pixel independently. Our analysis indicates that we can take a further step by simulating light transport in the latent space. To achieve this, the rendering process must be adapted to freely express the range of latent values observed.

5. Method

Our pilot study suggests that the latent representations might consist latent values producible by physically based rendering, and residual values that require additional modeling, e.g., high-frequency latents at shape boundaries. Based on these observations, we propose to disentangle the latents into two components: physically based latents and residual latents. We leverage light transport simulation to render the physically based latents, and use a neural refinement network to model the residual latents. This disentanglement allows us to separately address physically grounded effects and non-physically based phenomena present in the latents in a principled manner.

Our full pipeline is summarized in Figure 3. First, we apply differentiable rendering to train our physically based renderer such that it can output latent values by optimizing the scene parameters (π) to match reference latent images ($\tilde{I}_{\text{gt}}$), following our modified latent rendering formulation (Section 5.1). We then predict the residual latents using a neural refinement network (Section 5.3), which learns to compensate for discrepancies between the rendered latents and the target latents, particularly over geometric structures such as edges and corners. Details of our pipeline are as follows.

5.1. Latent rendering

As discussed in the latent analysis section (Section 4), the latent space includes both positive and negative values, whereas conventional physically based rendering produces only non-negative values. A naïve way to adapt a physically based renderer for signed rendering is to shift latent values into the non-negative domain by applying an offset and a scale on the latents, or by splitting the positive and negative latents and perform negation on the negatives. However, we find that both approaches are not effective and efficient to address the observations as outlined in Section 4. Particularly, shifting the values will result in appearance changes while splitting the latents requires doubling the number of latent channels. Reconstructing the modified latents is challenging as well as the decoder requires to de-normalize the latents to their vanilla domain before decoding. Instead of processing the latents heuristically, we therefore opt for a principled approach by modeling the latents using the rendering equation. Let us begin by introducing the concept of *signed rendering* as follows.

5.1.1. Signed radiance and reflectance

In keeping with our goal of strong physically-based priors, we begin our latent rendering approach by modeling the latents' physically-aligned features: structure, shading, and global illumination. In traditional rendering, emitted light L_e and the BSDF f_r are constrained to be non-negative in order to ensure valid light path sampling. To accommodate signed latent values, we modify the traditional path tracing approach to support the propagation of both positive and negative energy throughout the scene. First, we allow emitters to emanate signed light energy $\tilde{L}_e(\mathbf{x} \leftarrow \mathbf{y}, \pi)$, so that $\tilde{L}_e$ can emit positive or negative radiance values. We also modify BSDF evaluations to enable them to yield signed values. This leads to the following signed light transport equation:

$$\tilde{I}_i^e(\pi) = \int_{\mathcal{P}} r_i(\tilde{\mathbf{x}}) \tilde{T}(\tilde{\mathbf{x}}, \pi) \tilde{L}_e(\mathbf{x}_{k-1} \leftarrow \mathbf{x}_k, \pi) d\tilde{\mathbf{x}}, \quad (4)$$

where $\tilde{T}$ is the signed throughput. We note that purely specular materials are not modified, as they behave strictly as ideal mirror surfaces.

With both signed radiance and signed reflectance enabled, we can effectively model the observed behavior that latents can take on values outside of the standard rendering range while still maintaining the structure of the shading and lighting effects present in the original image. Our subsequent modifications use this signed rendering as a starting point, and aim to model the rendered latents in a way that will capture features that deviate from a physical basis.

5.1.2. Flat response term

Our latent analysis indicates that, in some channels, different surfaces exhibit a comparatively flat appearance rather than faithfully reflecting their original material properties. To account for this behavior, we introduce an additional flat response term applied at surface interactions, which provides explicit control over the extent to which objects preserve material-dependent variation versus collapsing to a near-constant response. The flat response term $\tilde{L}_a$ is a signed radiance value, and its contribution can be estimated with:

$$\tilde{I}_i^a(\pi) = \int_{\mathcal{P}_E} r_i(\tilde{\mathbf{x}}) \tilde{T}(\tilde{\mathbf{x}}, \pi) \tilde{L}_a(\mathbf{x}_d, \pi) d\tilde{\mathbf{x}}. \quad (5)$$

where the path space $\mathcal{P}_E$ denotes a camera subpath $\tilde{\mathbf{x}}$ following the ES^*D format, i.e., a path starting at the camera sensor (E) with an optional series of specular reflections (S^*) followed by a diffuse interaction (D). Note that it is valid to define the flat-response term for full light transport paths, by accumulate it at each interaction, following the original path space $\mathcal{P}$; however, we empirically observe that this does not lead to a significant performance difference, and hence opt for the simpler design using only camera subpaths.

5.1.3. Occlusion term

Our latent analysis also shows that shaded regions can deviate from the physically based interpretation of light transport, exhibiting higher intensity than unoccluded regions of the same material, or even values of opposite sign. To model this behavior, we introduce an occlusion term that acts as an additional contribution emitted only from shaded regions. The occlusion term $\tilde{L}_c$ stores signed radiance values, and its contribution to the rendered intensity is estimated as

$$\tilde{I}_i^c(\pi) = \int_{\mathcal{P}_E} r_i(\tilde{\mathbf{x}}) \tilde{T}(\tilde{\mathbf{x}}, \pi) \tilde{L}_c(\mathbf{x}_d, \pi) V(\mathbf{x}_d, \mathbf{y}) d\tilde{\mathbf{x}}. \quad (6)$$

Here, we leverage camera subpaths $\tilde{\mathbf{x}} \in \mathcal{P}_E$ with an ending diffuse vertex $\mathbf{x}_d$ to evaluate this integral. We sample an additional light vertex $\mathbf{y}$ on a light source and evaluate the binary visibility function $V(\mathbf{x}_d, \mathbf{y})$, which equals one when the ending diffuse vertex $\mathbf{x}_d$ is visible from $\mathbf{y}$, and zero otherwise.

5.1.4. Latent rendering equation

By combining the above modifications, we obtain our latent rendering equation:

$$\tilde{I}_i(\pi) = \tilde{I}_i^e(\pi) + \tilde{I}_i^a(\pi) + \tilde{I}_i^c(\pi). \quad (7)$$

To estimate this latent intensity, we construct a single light path $\tilde{\mathbf{x}}$ using BSDF sampling and evaluate all corresponding integrals along that path. $\tilde{I}^a$ and $\tilde{I}^c$ are estimated when we encounter the first diffuse surface, whereas we continue to extend the path to estimate $\tilde{I}^e$. To efficiently estimating $\tilde{I}^e$, BSDF sampling is combined with next-event estimation (NEE) using multiple importance sampling, where the signed emitters ($\tilde{L}_e$) are explicitly sampled.

Note that rendering systems simulate lighting in linear radiance values, whereas generative models assume gamma-corrected inputs.

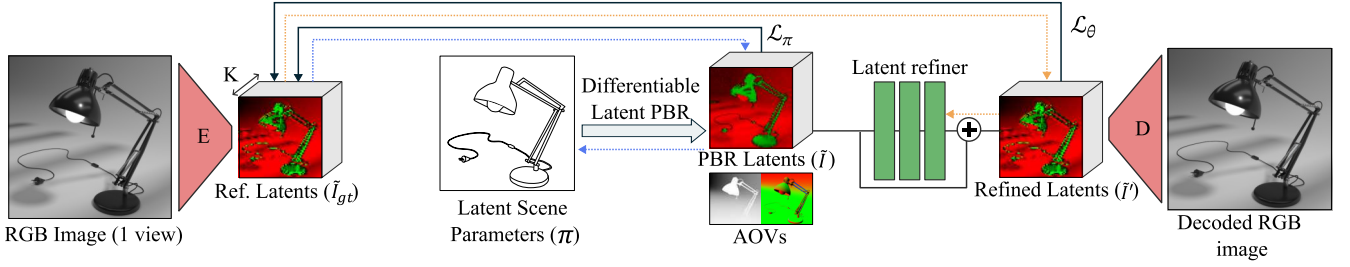


Figure 3: We train a differentiable renderer to output to the latent space defined by a variational autoencoder. We assume a 3D scene is available for our rendering task. Our training pipeline consists of taking a ground truth RGB rendering of the scene and encoding it into the latent space. We then optimize the 3D scene parameters to make its rendering reproduce this latent representation using our modified rendering equation (Eq. (7)). The rendered latents are then refined using a per-scene refiner network. The final latents are decoded to reconstruct the RGB rendering. Once the pipeline is trained, it can be applied to render latents for variants of the same scene with edited scene settings. The blue dotted line represents backpropagation for scene parameter optimization. The yellow dotted line represents backpropagation for the refiner parameter training.

For this reason, we apply gamma correction with $\gamma = 2.2$ to the positive and negative values of the rendered latents, respectively:

$$g(x, \gamma) = \max(x, 0)^\gamma - \max(-x, 0)^\gamma. \quad (8)$$

5.2. Optimization

We use a differentiable renderer [JSRV22] to perform latent rendering (Eq. (7)). We optimize the scene parameters to make the renderer produce latents following the latent space defined by the VAE encoder. All latent channels are jointly rendered, analogously to RGB rendering. For a given scene and a reference latent map from a training view, we optimize the parameters $\pi = \{\tilde{L}_e, \tilde{L}_a, \tilde{L}_c\}$ for each path vertex, which is represented as a vector in $\mathbb{R}^d$ with d as the total latent channels. We also optimize the reflectance parameters of each BSDF present in the scene, as well as the alpha parameters of any rough BSDFs. We support diffuse materials, conductors, dielectrics, and plastics and their rough variants. We apply a clipping function to make sure the learnable parameters are non-negative during optimization.

To train the renderer, we minimize a Huber loss between the gamma-corrected rendered latents $\tilde{I}$ and ground-truth latents I_{gt} :

$$\mathcal{L}_\pi = \begin{cases} \frac{1}{2} \|\tilde{I} - I_{gt}\|_2^2 & \text{if } \|\tilde{I} - I_{gt}\|_1 \leq \delta \\ \delta \left(\|\tilde{I} - I_{gt}\|_1 - \frac{1}{2} \delta \right) & \text{if } \|\tilde{I} - I_{gt}\|_1 > \delta \end{cases} \quad (9)$$

We set $\delta = 1$, which can make optimization more stable when faced with large but infrequent per-pixel errors due to the strongly encoded edges observed in the pilot study. Note that we compute the loss directly in the latent space, which empirically improves training convergence, stability, and efficiency. This also helps avoid inefficient gradient backpropagation through the decoder.

5.3. Latent refinement

The rendered latents produced by our fitted scene parameters and latent rendering equation impose a strong physically based prior. However, as discussed in the latent analysis, latent values can

deviate from physically based behavior, particularly around object boundaries and other high-frequency regions. We train a feed-forward neural network to predict these residual latents. For simplicity, we train this neural refiner on a per-scene basis using a single ground truth rendering. We leave the generalization of the refiner across scenes for future work.

The refiner network R_θ takes as input the rendered latent $\tilde{I}$ along with a set of auxiliary buffers (AOVs) denoted as A . These AOVs include surface shading normals and depth, providing geometric cues that guide the network toward correcting structure-dependent artifacts. The AOVs are rendered at the latent resolution and concatenated to the input in a channel-wise manner. The network outputs the residual latents as $R_\theta(\tilde{I}, A)$.

We adopt a feed-forward multilayer perceptron architecture for the refiner network. The hidden layer width is chosen as a multiple of the number of latent channels d . For Stable Diffusion 3.5, we use an expansion factor of 16, resulting in hidden layers with $16 \times 16 = 256$ channels. The network then consists of three convolution blocks with kernel sizes 3×3 , 5×5 , and 3×3 .

We train the refiner in a separate stage after the physically based renderer is optimized. To train the refiner, we use a composite loss defined in both the latent and RGB spaces:

$$\mathcal{L}_\theta = \mathcal{L}_{MSE}(\tilde{I}', \tilde{I}_{gt}) + \mathcal{L}_{SSIM}(\tilde{I}', \tilde{I}_{gt}) + \mathcal{L}_{LPIPS}(\mathcal{D}(\tilde{I}'), \mathcal{D}(\tilde{I}_{gt})), \quad (10)$$

where the refined latent is $\tilde{I}' = \tilde{I} + R_\theta(\tilde{I}, A)$ and $\mathcal{D}$ denotes the decoder. The MSE loss and SSIM loss enforces the refined latents to stay close to the ground-truth latents. The LPIPS loss [ZIE*18] applies perceptual similarity constraints on the decoded image, reducing color shifts and encouraging sharper boundaries.

6. Experimental Results

6.1. Implementation details

We use Mitsuba [JSRV22] with path replay backpropagation [VSJ21], without reparameterization [ZRJ23], to perform the

optimization. We extend Mitsuba to support signed rendering with 16 latent channels, enabling efficient joint optimization over all channels. We also support textured materials: each RGB reflectance texture is converted to grayscale and then mapped into the 16-channel latent space via a learnable per-channel scale and offset. This preserves the texture’s spatial detail while exposing its latent appearance as optimizable parameters. Training follows a multi-stage procedure: we first optimize the scene parameters for 3000 iterations with a learning rate of 10^{-4} , and then train the refinement network for an additional 3000 iterations using the same learning rate. All optimizations use the Adam optimizer [KB15] with default hyperparameters. We plan to release our implementation upon acceptance.

To improve stability during scene parameter optimization, we employ a learning-rate scheduler following a cosine annealing schedule with restarts, as well as early stopping. On a simple scene such as the Cornell Box, training on a single NVIDIA RTX 4090 GPU takes 5 minutes for scene optimization and 5 minutes for refiner training. Scene parameters π are initialized randomly from a uniform distribution in $[0, 1]$ for both positive and negative components of all trainable parameters, as well as the BSDF parameters.

We compare our method with a baseline that uses the traditional rendering formula [Kaj86] for latent rendering. We produce 16-channel values in the $[0, 1]$ range with the traditional rendering process, then de-normalize the rendered latents back into the native latent range $\sim [-5, 5]$ using the logit function $\log(x/(1-x))$, and train the refiner on these de-normalized latents following Section 5.3. The refiner output is passed directly to the decoder. Compared to our method, this baseline skips the latent modeling in Section 5.1 and simply uses the traditional rendering formula.

6.2. Scene setups

We analyze the performance of our latent rendering method on five scenes of increasing complexity: Cornell Box, Lamp, Veach-Bidir, Dining Room, and Living Room. The Cornell Box scene contains a single light source and a mix of diffuse and pure specular materials. The Lamp scene includes multiple light sources and rough plastic materials. The Veach-Bidir scene contains difficult-to-sample light sources and caustics effects. The Dining Room and Living Room scene features a single light source and multiple objects with distinct BSDFs. For Cornell Box and Veach-Bidir, we associate each object with a separate BSDF with its own learnable parameters. For Lamp, Dining Room, Living Room scenes, we use the default shape-BSDF associations provided in original scenes to reduce memory consumption during the optimization of the BSDF parameters. We assign each object with a non-delta BSDF its own flat-response and occlusion parameters, $\tilde{L}_a$ and $\tilde{L}_c$, which have random initializations. For each scene, we provide reference latents obtained from a single view. These reference latents are obtained by encoding the RGB rendering produced with path tracing on large number of samples per pixels (e.g., 16K spps).

We demonstrate the capabilities of our latent rendering framework on scene editing tasks. In all experiments, the scene geometry, material types (e.g., diffuse or specular), emitter locations, and camera pose are assumed to be known. We perform scene edits and render latents in the edited scene to explore the generalization of latent

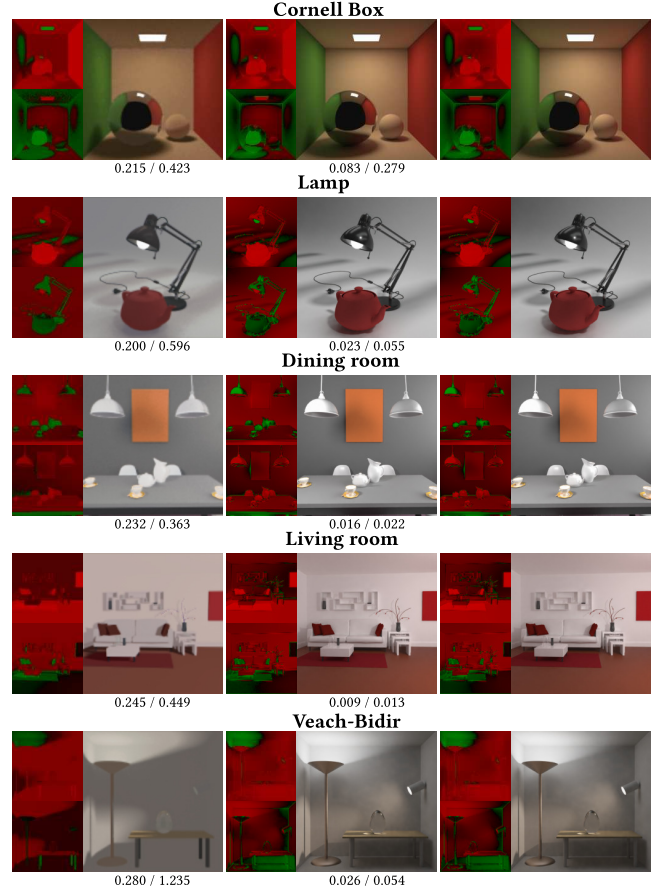


Figure 4: We compare the performance of the training view on rendered latent and refined latent with the ground truth. Representative latents for each case are shown on the left of each color rendering. Metrics shown are LPIPS/MSEx100.

rendering to various scenarios. We consider camera movement for novel view synthesis, light source movement for scene relighting, and object movement for animated rendering. Relaxing the scene input assumptions such as taking a single image as input and predicting scene geometry and PBR materials [WXD*25, XCX*25] would be an interesting direction for future work. We compare the latents and their reconstruction with ground truth latents and images, respectively. We report comparison results using the MSE, SSIM [WBSS04], and LPIPS metrics.

6.3. Qualitative results

Fig. 4 demonstrates how our method performs before and after refinement on the training view. For completeness, we show two sets of representative latents for each decoded RGB image. We observe, across all scenes, that the main content of the latents and their decoded images is produced by the latent renderer, as evidenced by their high similarity to the ground truth. The refiner is then able to overfit and deliver an almost perfect reconstruction.

Figs. 7 and 11 provide the rendering results when the scene is

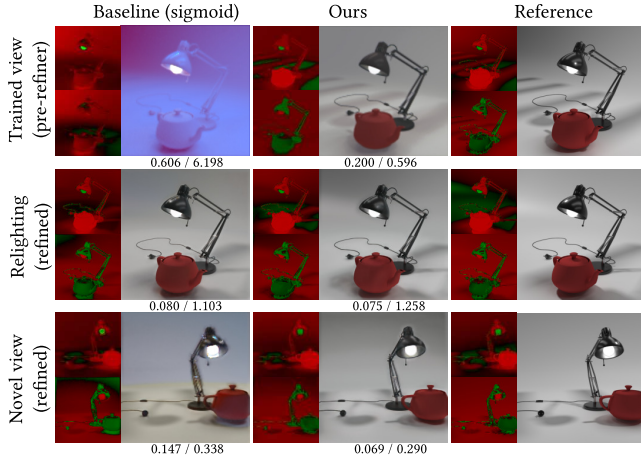


Figure 5: Comparison between the baseline and our method on the Lamp scene. We observe that our latent modeling and rendering formula provide significant quality improvement compared to the baseline that only relies on latent normalization and traditional rendering. Metrics shown are LPIPS/MSE $\times 100$.

edited in various scenarios. We can observe that our method generalizes well to novel configurations: most of the work is done by the rendered latents, while the refiner is responsible for improving object boundaries and increasing overall image contrast and sharpness.

For the training view, we found that the post-refinement reconstruction is almost indistinguishable from the ground truth image, which confirms the success of the training. When the scene is edited, through camera motion, changes in light position, or object movement, our method can regenerate images that remain visually close to the ground truth. This result demonstrates two advantages of latent rendering: the trained renderer can be applied to synthesize new latents that can be reconstructed to meaningful images, and the refiner generalizes to unseen settings. Notably, specular reflections and shadowing effects can be reconstructed based on the visual cues in the physically based rendered latents.

Fig. 12 shows novel view rendering on three additional scenes (Bedroom, Classroom, and Living Room 2) with texture support. It shows that latent rendering with texture is promising, with further optimizations for quality improvement left as future work. Video results for each scene are provided in the supplemental material.

Baseline comparison. We provide baseline comparisons in Fig. 5. Compared to our method, the rendered latents of the baseline yield color shifts in the reconstruction (1st row). This could be explained by the non-linear range compression and expansion due to normalization and de-normalization, respectively. The refiner significantly improves the rendered latents and the final rendering, but fails to achieve accurate color rendering while having more visual artifacts in the baseline (2nd and 3rd rows). This result confirms the necessity and effectiveness of our latent rendering equation.

Errors of scene edits. Fig. 6 provides the error plot of the scene edits for the Lamp scene. The edits are performed for 60 frames,

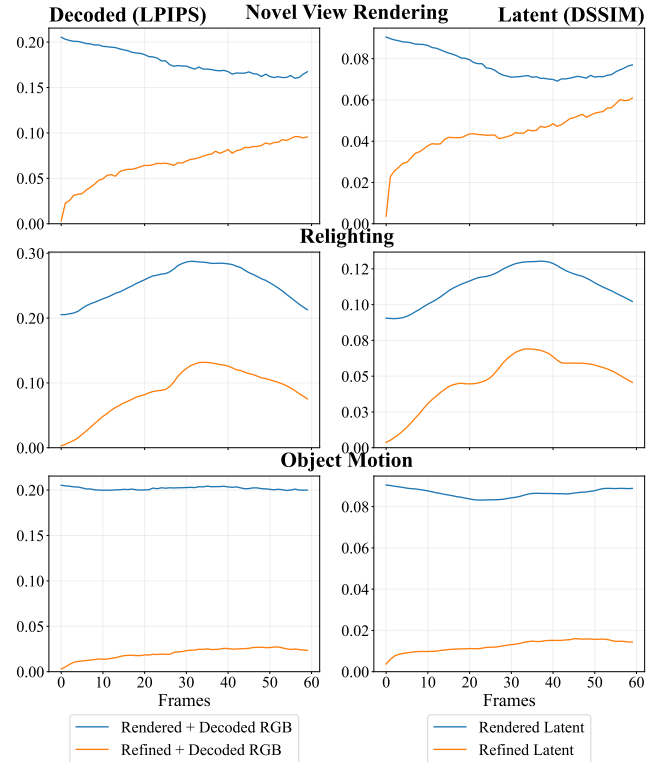


Figure 6: The error of the Lamp scene across 60 frames of scene editing variants. Frame 0 corresponds to the training view.

starting from frame 0 as the training view. We use D-SSIM to measure structural difference and LPIPS to measure perceptual difference to the ground truth latents. For all tasks, the rendered latents generalize fairly well across frames with similar errors except for the extreme case in the light movement (around frame 30). The refiner improves the rendered latents significantly across frames. As the refiner is only trained on a single view, D-SSIM and LPIPS are smaller (i.e., better) at views more similar to the training view.

6.4. Quantitative results

Ablation studies. Table 1 presents our ablation study of the latent formulation and the impact of the refiner on the final decoded image across different scene edits. In this table, we report the mean LPIPS and MSE values attained along each animated editing sequence (moving camera, object and emitter). Using only signed emitters (Eq. (4)) leads to poor performance and unstable training. In contrast, the flat-response term (Eq. (5)) and occlusion term (Eq. (6)) have a significant impact on image quality. The neural refiner further improves the latents, enabling higher quality image reconstruction. Note that the refiner only acts on predicting the residual latents once the rendered latents are available. If the rendered latents are noisy (e.g., produced using only signed emitters), the refiner does not yield significant improvement.

Convergence analysis. Theoretically, latent rendering follows the Monte Carlo estimation framework, and hence, the convergence

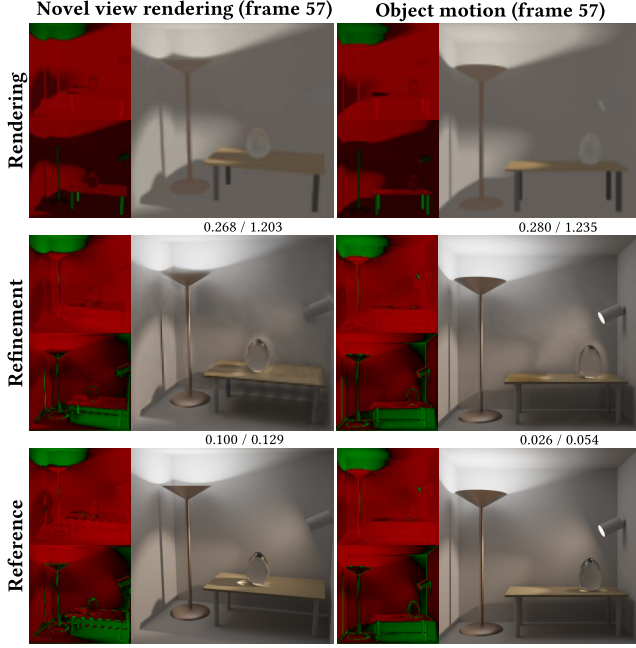


Figure 7: Qualitative results on the Veach-Bidir scene. Metrics shown are LPIPS/MSE $\times 100$.

Table 1: Ablation results reporting the mean LPIPS / MSE $\times 100$ achieved over the decoded RGBs. “Var.” denotes the scene-edit variant: camera, object, and light motion.

	Var.	$\tilde{I}_t^c$ (Eq. (4))	$+\tilde{I}_t^a$ (Eq. (5))	$+\tilde{I}_t^s$ (Eq. (6))	+ Refiner
Cornell Box	Cam.	0.502 / 2.264	0.228 / 0.335	0.213 / 0.222	0.075 / 0.069
	Obj.	0.526 / 2.355	0.213 / 0.338	0.202 / 0.261	0.032 / 0.061
	Light	0.519 / 2.361	0.209 / 0.330	0.191 / 0.220	0.013 / 0.024
Lamp	Cam.	0.394 / 3.091	0.206 / 0.844	0.178 / 0.564	0.068 / 0.297
	Obj.	0.439 / 3.274	0.236 / 0.791	0.202 / 0.608	0.020 / 0.048
	Light	0.504 / 5.702	0.292 / 3.000	0.253 / 2.385	0.086 / 1.867
Living Room	Cam.	0.643 / 5.502	0.281 / 0.599	0.250 / 0.480	0.072 / 0.115
	Obj.	0.648 / 5.584	0.280 / 0.573	0.246 / 0.451	0.008 / 0.013
	Light	0.659 / 6.031	0.299 / 1.053	0.266 / 0.894	0.064 / 0.392

properties in the latent space remains the same as in the RGB space. We use path tracing in this work, and therefore, the estimation of the rendered latents are unbiased. However, the use of the refiner network results in biased latents, and the image reconstruction using the decoder results in biased final rendering. Ideally, the rendering would become unbiased if the refiner and the decoder were linear functions. Empirically, we provide convergence plots of our rendering in Fig. 8. We observe that the convergence patterns are similar across scenes, and therefore provide the convergence of the Veach-Bidir scene as a representative. The plots confirm that, for the training view, both the latent and the decoded RGB estimates converge as the number of samples per pixel increases. The novel

view exhibits a similar trend, with a small residual bias introduced by the refiner.

6.5. Runtime analysis

We evaluate the runtime of latent rendering against RGB path tracing in an *equal-error* setting: for each method we measure the time and the number of samples per pixel required to reach a fixed MSE threshold (Fig. 9).

When the goal is to estimate values in the *latent space* (right), our method is significantly faster (42–84 $\times$) than first path tracing in RGB and then encoding, since the latent render and refiner are efficient and requires less samples. When the goal is a decoded *RGB image* (left), however, our method is not competitive with direct RGB path tracing: despite the spatial compression, the single decoder evaluation dominates the runtime (the red bar) and makes our method 6–35 $\times$ slower. Accelerating the decoder is therefore key to making latent rendering competitive for RGB output, and is an interesting direction for future work.

6.6. Applications

Scene editing. Latent rendering could be further explored for downstream applications. For example, as latent rendering naturally operates in the latent space, one can integrate score distillation sampling with latent rendering, enabling the optimization of scene parameters guided by a pre-trained diffusion prior [PJBM23]. Image editing applications could be integrated with latent rendering as well, e.g., object insertion and background removal using blended latents [AFL23], image harmonization and semantic image composition using delta denoising distillation [HACO23, CWY*24].

We provide an example of using score distillation [PJBM23] to re-color the teapot in the Lamp scene by optimizing its BSDF reflectance. The editing is guided by the text prompt “a blue teapot and a lamp on a table”. Traditional SDS operates the optimization in the latent space, and therefore it requires first performing the rendering in the RGB space, and then encode the rendered image using the encoder. By using latent rendering, we can directly compute the SDS objective based on the refined latents, skipping the encoder in the optimization pipeline. The results are shown in Fig. 10. Both the RGB rendering approach and our latent rendering approach can successfully update the teapot color. Due to the per-scene training of the refiner, our teapot results in some blurs that could be further corrected by improving the generalization of the refiner.

6.7. Limitations and future work

Our method is a very first step toward latent rendering. Several limitations remain for further investigations in future work. First, as the latent space is in low spatial resolutions, aliasing tends to dominate when rendering highly detailed objects. Second, while our refiner does fairly well at generalizing to intra-scene changes, it is trained only on a single image and with a per-scene setting, and thus does not generalize across scenes. Building a universal refiner across scenes would improve the practicality of the method. Finally, while latent rendering offers a new angle for extending physically based rendering research, there are still rooms for improving image

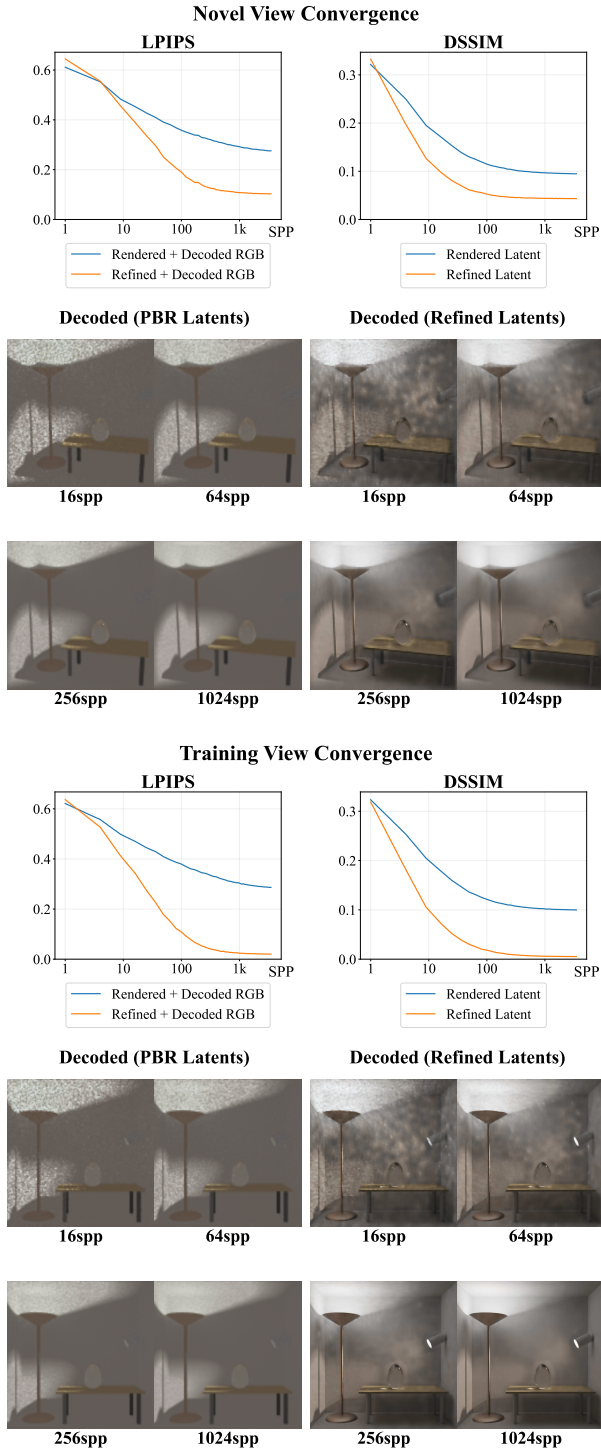


Figure 8: Empirical convergence vs. samples per pixel (spp) for a novel view (top) and a training view (bottom) of the Veach-Bidir scene. LPIPS on the decoded RGB (left) and DSSIM on the latents (right).

quality and running time to match the performance of traditional methods.

As future work, we aim to explore latent rendering with state-of-the-art visual diffusion models and pursue more accurate latent representations and processing for extended applicability and improved computational efficiency. For example, while our trained renderer and latent refiner generalizes well to higher resolution rendering, it is possible to add neural upsampling. Latent space super resolution [JHKK25] validated that since latent space represents image features in a spatially compressed format with high-level information, performing upsampling in this domain preserves well details and sharpness and could generate more details compared to RGB upsampling. Applying this method to improve our latent rendering would be future work.

7. Conclusion

We have presented latent rendering, a paradigm that trains a physically based renderer to output latent representations defined by the autoencoders of a generative model. The rendered latents can be reconstructed to color images following a neural refiner and an image decoder. Applications include direct editing of a 3D scene rendering in latent space, enabling a straightforward process to render scene variants under camera, lighting, and object changes. We believe that latent rendering is a promising approach to unlock the connection between physically based rendering and generative models. By leveraging latent spaces, we can enable the integration of physical priors directly into creative content generation.

Acknowledgment. We would like to thank Wig42 (DINING ROOM and LIVING ROOM), Jay-Artist (LIVING ROOM 2), No-vaZeeke (CLASSROOM), SlykDrako (BEDROOM), UP3D (LAMP), and Benedikt Bitterli [Bit16] for providing the scenes, which we obtained from the Mitsuba 3 gallery. This project is supported by Research Ireland under the Research Ireland Frontiers for the Future Programme - Project, award number 22/FFP-P/11522.

References

- [AFL23] AVRAHAMI O., FRIED O., LISCHINSKI D.: Blended latent diffusion. *ACM Trans. Graph.* (2023). doi:10.1145/3592450. 9
- [BFB*25] BHARADWAJ S., FENG H., BECHERINI G., FERNANDEZ ABBEY V., BLACK M. J.: Genlit: Reformulating single-image relighting as video generation. In *ACM SIGGRAPH Asia Conference Papers* (2025). doi:10.1145/3757377.3763970. 3
- [Bit16] BITTERLI B.: Rendering resources. <https://benedikt-bitterli.me/resources/>, 2016. 10
- [Bla25] BLACK FOREST LABS: FLUX.2: Analyzing and enhancing the latent space of FLUX — representation comparison, 2025. URL: [https://bfl.ai/research/representation-comparison. 2, 4](https://bfl.ai/research/representation-comparison.2,4)
- [Ble25] BLENDER FOUNDATION: Blender – a 3d modelling and rendering package. <https://www.blender.org>, 2025. 2
- [BPE17] BAUSZAT P., PETITJEAN V., EISEMANN E.: Gradient-domain path reusing. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)* 36, 6 (2017), 229:1–229:9. doi:10.1145/3130800.3130886. 2
- [BSH02] BEKAERT P., SBERT M., HALTON J.: Accelerating path tracing by re-using paths. In *Proceedings of the Eurographics Workshop*

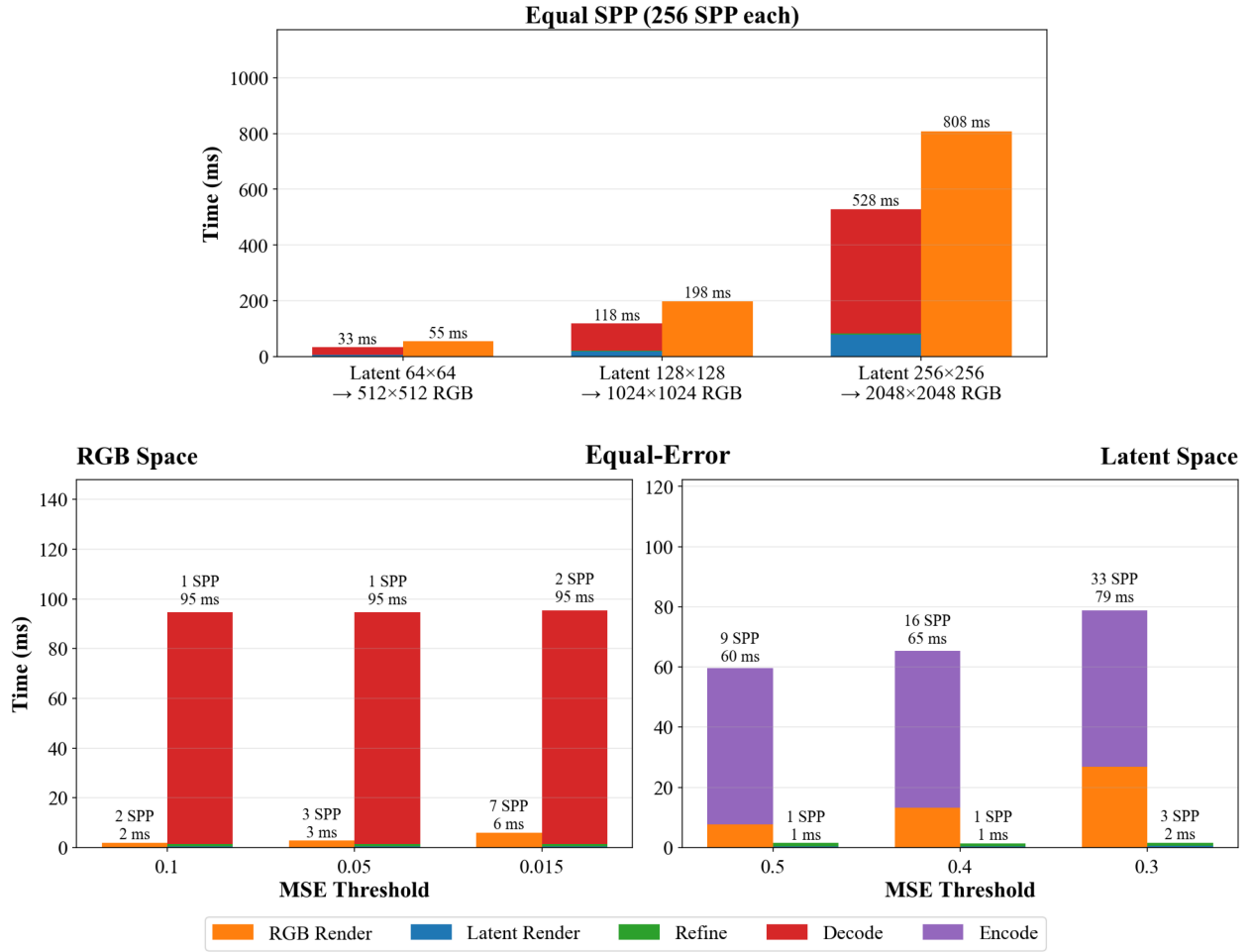


Figure 9: Equal-error runtime comparison between RGB path tracing and our latent rendering method (with refiner), for RGB space (left) and latent space (right). The plot is based on frame 30 of the move_camera test case of the Lamp scene.

- on Rendering (2002), pp. 125–134. doi:10.2312/EGWR/EGWR02/125-134. 2
- [CA25] CAREAGA C., AKSOY Y.: Physically controllable relighting of photographs. In *ACM SIGGRAPH Conference Papers* (2025). doi:10.1145/3721238.3730666. 3
- [CWY*24] CHEN Z., WANG W., YANG Z., YUAN Z., CHEN H., SHEN C.: Freecompose: Generic zero-shot image composition with diffusion prior. In *ECCV* (2024). doi:10.1007/978-3-031-72643-9_5. 9
- [EKB*24] ESSER P., KULAL S., BLATTMANN A., ENTEZARI R., MÜLLER J., SAINI H., LEVI Y., LORENZ D., SAUER A., BOESEL F., ET AL.: Scaling rectified flow transformers for high-resolution image synthesis. In *International Conference on Machine Learning (ICML)* (2024). 4
- [HACO23] HERTZ A., ABERMAN K., COHEN-OR D.: Delta denoising score. In *IEEE/CVF International Conference on Computer Vision (ICCV)* (2023). doi:10.1109/ICCV51070.2023.00221. 9
- [HGP*19] HUA B.-S., GRUSON A., PETITJEAN V., ZWICKER M., NOWROUZEZHAI D., EISEMANN E., HACHISUKA T.: A survey on gradient-domain rendering. *Computer Graphics Forum (Proceedings of Eurographics State of the Art Reports)* 38, 2 (2019), 455–472. doi:10.1111/cgf.13652. 2
- [HJA20] HO J., JAIN A., ABBEEL P.: Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)* (2020). 2, 3
- [HOJ08] HACHISUKA T., OGAKI S., JENSEN H. W.: Progressive photon mapping. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)* (2008). doi:10.1145/1409060.1409083. 2
- [HSHS*26] HIRSCHORN O., SELA O., HUBERMAN-SPIEGELGLAS I., EFRAT N., ALSHAN E., IDESES I., DEVERNAVY F., ZVIK Y., FRITZ L.: Splatent: Splatting diffusion latents for novel view synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2026), pp. 8319–8330. arXiv:2512.09923. 3
- [HSRM25] HERHOLZ S., SIK M., REICHARDT L., MANZI M.: Path guiding in production and recent advancements. In *ACM SIGGRAPH Courses* (2025). doi:10.1145/3721241.3733994. 2
- [HSW*22] HU E. J., SHEN Y., WALLIS P., ALLEN-ZHU Z., LI Y., WANG S., WANG L., CHEN W.: LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)* (2022). 3
- [HWH*25] HE Z., WANG T., HUANG X., PAN X., LIU Z.: Neural

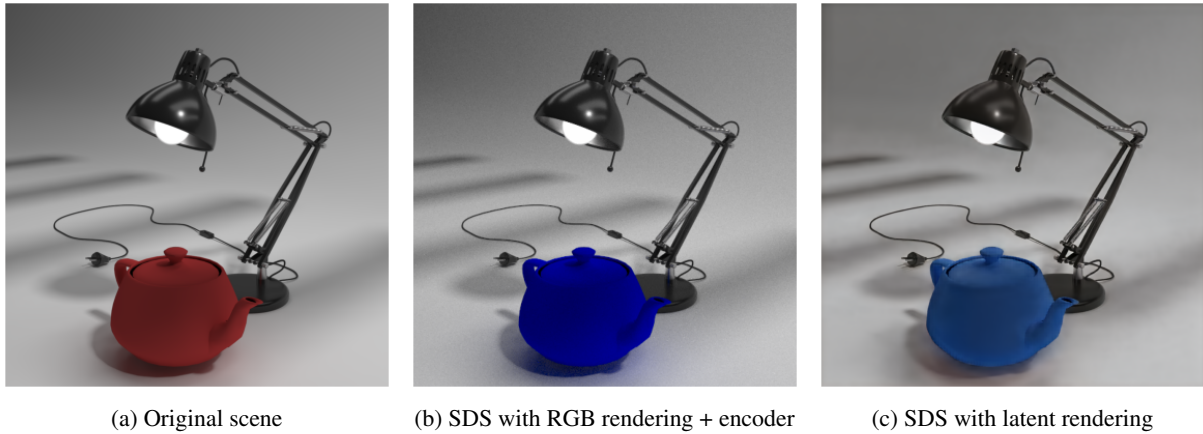


Figure 10: Application using score distillation sampling (SDS) [PJB23]. We use the pretrained diffusion model to iteratively update the reflectance of the teapot following guidance from the text prompt “A blue teapot and a lamp on a table.”

- LightRig: Unlocking accurate object normal and material estimation with multi-light diffusion. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2025), pp. 26514–26524. doi:10.1109/CVPR52734.2025.02469. 3
- [Jen96] JENSEN H. W.: Global illumination using photon maps. In *Proceedings of the Eurographics Workshop on Rendering* (1996), Springer. doi:10.1007/978-3-7091-7484-5_3. 2
- [JHKK25] JEONG J., HAN S., KIM J., KIM S. J.: Latent space super-resolution for higher-resolution image generation with diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference* (2025), pp. 2355–2365. doi:10.1109/CVPR52734.2025.00225. 10
- [JSRV22] JAKOB W., SPEIERER S., ROUSSEL N., VICINI D.: Dr.jit: A just-in-time compiler for differentiable rendering. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* (2022). doi:10.1145/3528223.3530099. 6
- [Kaj86] KAJIYA J. T.: The rendering equation. In *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques* (1986), SIGGRAPH ’86, ACM. doi:10.1145/15922.15902. 1, 2, 7
- [KB15] KINGMA D. P., BA J.: Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)* (2015). arXiv:1412.6980. 3, 7
- [KHN25] KOCIS P., HÖLLEIN L., NIESSNER M.: IntrinsicX: High-quality PBR generation using image priors. In *Advances in Neural Information Processing Systems (NeurIPS)* (2025). 3
- [KMA*15] KETTUNEN M., MANZI M., AITTALA M., LEHTINEN J., DURAND F., ZWICKER M.: Gradient-domain path tracing. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* (2015). doi:10.1145/2766997. 2
- [KW14] KINGMA D. P., WELING M.: Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)* (2014). 4
- [LADL18] LI T.-M., AITTALA M., DURAND F., LEHTINEN J.: Differentiable monte carlo ray tracing through edge sampling. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)* (2018). doi:10.1145/3272127.3275109. 3
- [LDHG*25] LYU L., DESCHAMPTRE V., HOLD-GEOFFROY Y., HAŠAN M., YOON J. S., LEIMKÜHLER T., THEOBALT C., GEORGIEV I.: IntrinsicEdit: Precise generative image manipulation in intrinsic space. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* (2025). doi:10.1145/3731173. 3
- [LGL*25] 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2025). doi:10.1109/CVPR52734.2025.02428. 3
- [LKL*13] LEHTINEN J., KARRAS T., LAINE S., AITTALA M., DURAND F., AILA T.: Gradient-domain metropolis light transport. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* 32, 4 (2013), 95:1–95:12. doi:10.1145/2461912.2461943. 2
- [LW93] LAFORTUNE E. P., WILLEMS Y. D.: Bi-directional path tracing. In *Proceedings of Compugraphics* (1993). 2
- [LWVH*23] LIU R., WU R., VAN HOORICK B., TOKMAKOV P., ZAKHAROV S., VONDRICK C.: Zero-1-to-3: Zero-shot one image to 3d object. In *IEEE/CVF International Conference on Computer Vision (ICCV)* (2023). doi:10.1109/ICCV51070.2023.00853. 3
- [MHS*22] MENG C., HE Y., SONG Y., SONG J., WU J., ZHU J.-Y., ERMON S.: SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations (ICLR)* (2022). 3
- [MRP*23] METZER G., RICHARDSON E., PATASHNIK O., GIRYES R., COHEN-OR D.: Latent-nerf for shape-guided generation of 3d shapes and textures. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2023). doi:10.1109/CVPR52729.2023.01218. 3
- [NDSRJ20] NIMIER-DAVID M., SPEIERER S., RUIZ B., JAKOB W.: Radiative backpropagation: An adjoint method for lightning-fast differentiable rendering. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* (2020). doi:10.1145/3386569.3392406. 3
- [NDVZJ19] NIMIER-DAVID M., VICINI D., ZELTNER T., JAKOB W.: Mitsuba 2: A retargetable forward and inverse renderer. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)* (2019). doi:10.1145/3355089.3356498. 3
- [PJB23] POOLE B., JAIN A., BARRON J. T., MILDENHALL B.: Dreamfusion: Text-to-3d using 2d diffusion. In *International Conference on Learning Representations (ICLR)* (2023). 2, 9, 12
- [PJH23] PHARR M., JAKOB W., HUMPHREYS G.: *Physically Based Rendering: From Theory to Implementation*, 4 ed. MIT Press, 2023. 2
- [RBL*22] ROMBACH R., BLATTMANN A., LORENZ D., ESSER P., OMER B.: High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022). doi:10.1109/CVPR52688.2022.01042. 2, 3, 4

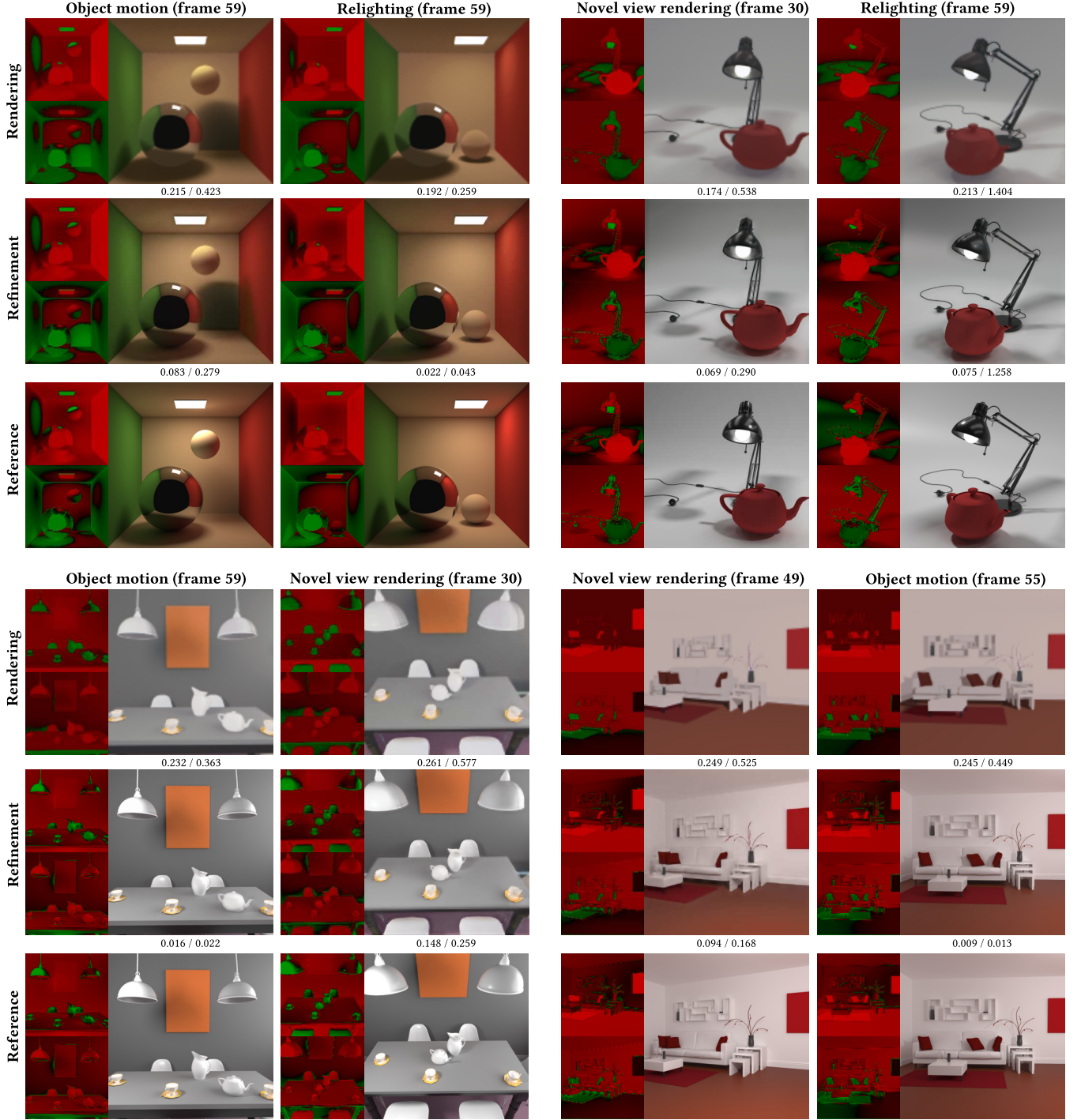


Figure 11: We compare the performance of the scene editing on rendered latent and refined latent with the ground truth. Representative latents for each case are shown on the left of each color rendering. Metrics shown are LPIPS/MSE $\times 100$.



Figure 12: We compare the performance of novel view scene editing on rendered latents and refined latents with the ground truth. We showcase complex scenes with texture support. Metrics shown are LPIPS/MSe $\times 100$.

- [Rey03] REYNOLDS O.: *Papers on Mechanical and Physical Subjects: The Sub-Mechanics of the Universe*, vol. 3. The University Press, 1903. 3
- [SDWMG15] SOHL-DICKSTEIN J., WEISS E., MAHESWARANATHAN N., GANGULI S.: Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning (ICML)* (2015). 2, 3
- [SE19] SONG Y., ERMON S.: Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems (NeurIPS)* (2019). 2, 3
- [SPM25] SOROKA M., PETERS C., MARSCHNER S.: Quadric-based silhouette sampling for differentiable rendering. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* (2025). doi:10.1145/3731146. 3
- [Tur22] TURNER K.: Decoding latents to RGB without up-scaling. <https://discuss.huggingface.co/t/decoding-latents-torgb-without-upscaling/23204>, 2022. Hugging Face Forums. 4
- [Vea97] VEACH E.: *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford University, 1997. 3
- [VG95] VEACH E., GUIBAS L. J.: Bidirectional estimators for light transport. In *Photorealistic Rendering Techniques*. Springer, 1995. doi:10.1007/978-3-642-87825-1_11. 2
- [VG97] VEACH E., GUIBAS L. J.: Metropolis light transport. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques* (1997), SIGGRAPH '97, ACM. doi:10.1145/258734.258775. 2
- [VSJ21] VICINI D., SPEIERER S., JAKOB W.: Path replay backpropagation: Differentiating light paths using constant memory and linear time. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* (2021). doi:10.1145/3450626.3459804. 3, 4, 6
- [WBSS04] WANG Z., BOVIK A. C., SHEIKH H. R., SIMONCELLI E. P.: Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing* 13, 4 (2004), 600–612. doi:10.1109/TIP.2003.819861. 7
- [WKL*23] WYMAN C., KETTUNEN M., LIN D., BITTERLI B., YUKSEL C., JAROSZ W., KOZLOWSKI P., DE FRANCESCO G.: A gentle introduction to ReSTIR: Path reuse in real-time. In *ACM SIGGRAPH Courses* (2023). doi:10.1145/3587423.3595511. 2
- [WWA*25] WAN T., WANG A., AI B., WEN B., MAO C., XIE C.-W., CHEN D., YU F., ZHAO H., YANG J., ZENG J., WANG J., ZHANG J., ZHOU J., WANG J., CHEN J., ZHU K., ZHAO K., YAN K., HUANG L., FENG M., ZHANG N., LI P., WU P., CHU R., FENG R., ZHANG S., SUN S., FANG T., WANG T., GUI T., WENG T., SHEN T., LIN W., WANG W., WANG W., ZHOU W., WANG W., SHEN W., YU W., SHI X., HUANG X., XU X., KOU Y., LV Y., LI Y., LIU Y., WANG Y., ZHANG Y., HUANG Y., LI Y., WU Y., LIU Y., PAN Y., ZHENG Y., HONG Y., SHI Y., FENG Y., JIANG Z., HAN Z., WU Z.-F., LIU Z.: Wan: Open and advanced large-scale video generative models, 2025. arXiv:2503.20314. 2, 3
- [WXD*25] WANG R., XU S., DAI C., XIANG J., DENG Y., TONG X., YANG J.: MoGe: Unlocking accurate monocular geometry estimation for open-domain images with optimal training supervision. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2025). doi:10.1109/CVPR52734.2025.00496. 3, 7
- [XBLZ24] XU P., BANGARU S., LI T.-M., ZHAO S.: Markov-chain monte carlo sampling of visibility boundaries for differentiable rendering. In *ACM SIGGRAPH Asia Conference Papers* (2024). doi:10.1145/3680528.3687622. 3
- [XCC*25] XIE E., CHEN J., CHEN J., CAI H., TANG H., LIN Y., ZHANG Z., LI M., ZHU L., LU Y., HAN S.: Sana: Efficient high-resolution image synthesis with linear diffusion transformers. In *International Conference on Learning Representations (ICLR)* (2025). arXiv:2410.10629. 2, 3
- [XCX*25] XIANG J., CHEN X., XU S., WANG R., LV Z., DENG Y., ZHU H., DONG Y., ZHAO H., YUAN N. J., YANG J.: Native and compact structured latents for 3d generation, 2025. arXiv:2512.14692. 7
- [YLWX25] YAN D., LI Z., WU L., XU K.: Generalized unbiased reconstruction for gradient-domain rendering. *ACM Trans. Graph.* 44, 6 (2025). doi:10.1145/3763297. 2
- [YTZ*25] YANG Z., TENG J., ZHENG W., DING M., HUANG S., XU J., YANG Y., HONG W., ZHANG X., FENG G., ET AL.: Cogvideox: Text-to-video diffusion models with an expert transformer. In *International Conference on Learning Representations (ICLR)* (2025). arXiv:2408.06072. 3
- [ZCZ25] ZENG Y., CAI G., ZHAO S.: A survey on physics-based differentiable rendering, 2025. arXiv:2504.01402. 3
- [ZDG*24] ZENG Z., DESCHAMPTRE V., GEORGIEV I., HOLD-GEORFFROY 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 Conference Papers* (2024). doi:10.1145/3641519.3657445. 3
- [ZDP*25] ZENG C., DONG Y., PEERS P., WU H., TONG X.: Ren-derformer: Transformer-based neural rendering of triangle meshes with global illumination. In *ACM SIGGRAPH Conference Papers* (2025). doi:10.1145/3721238.3730595. 3
- [ZGV*25] ZHOU J. J., GAO H., VOLETI V., VASISHTA A., YAO C.-H., BOSS M., TORR P., RUPPRECHT C., JAMPANI V.: Stable virtual camera: Generative view synthesis with diffusion models. In *IEEE/CVF International Conference on Computer Vision (ICCV)* (2025), pp. 12405–12414. doi:10.1109/ICCV51701.2025.01153. 3
- [ZIE*18] ZHANG R., ISOLA P., EFROS A. A., SHECHTMAN E., WANG O.: The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2018). doi:10.1109/CVPR.2018.00068. 6
- [ZMY*20] ZHANG C., MILLER B., YAN K., GKIIOULEKAS I., ZHAO S.: Path-space differentiable rendering. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* (2020). doi:10.1145/3386569.3392383. 3
- [ZRA23] ZHANG L., RAO A., AGRAWALA M.: Adding conditional control to text-to-image diffusion models. In *IEEE/CVF International Conference on Computer Vision (ICCV)* (2023). doi:10.1109/ICCV51070.2023.00355. 3
- [ZRJ23] ZHANG Z., ROUSSEL N., JAKOB W.: Projective sampling for differentiable rendering of geometry. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)* (2023). doi:10.1145/3618385. 3, 6