ROGR: Relightable 3D Objects using Generative Relighting

Jiapeng Tang[1,3*] **Matthew Levine**[1*] **Dor Verbin**[2] **Stephan J. Garbin**[1]
Matthias Nießner[3] **Ricardo Martin-Brualla**[1] **Pratul P. Srinivasan**[2] **Philipp Henzler**[1]
[1] Google Research [2] Google Deepmind [3] Technical University of Munich

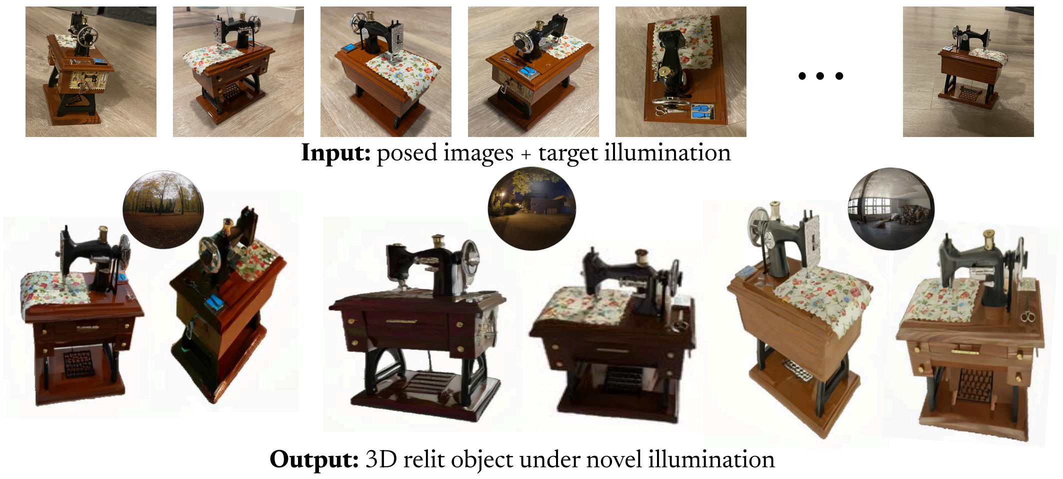

Input: posed images + target illumination
Output: 3D relit object under novel illumination

Figure 1: Given a set of posed images under unknown illumination (top), our method reconstructs a relightable neural radiance field (bottom), that can be rendered under any novel environment map without further optimization, on-the-fly relighting and novel view synthesis.

Abstract

We introduce ROGR, a novel approach that reconstructs a relightable 3D model of an object captured from multiple views, driven by a generative relighting model that simulates the effects of placing the object under novel environment illuminations. Our method samples the appearance of the object under multiple lighting environments, creating a dataset that is used to train a lighting-conditioned Neural Radiance Field (NeRF) that outputs the object's appearance under any input environmental lighting. The lighting-conditioned NeRF uses a novel dual-branch architecture to encode the general lighting effects and specularities separately. The optimized lighting-conditioned NeRF enables efficient feed-forward relighting under arbitrary environment maps without requiring per-illumination optimization or light transport simulation. We evaluate our approach on the established TensoIR and Stanford-ORB datasets, where it improves upon the state-of-the-art on most metrics, and showcase our approach on real-world object captures. Project Page

1 Introduction

Inserting real-world objects into new environments is a long-standing problem in computer graphics [1, 2], with numerous applications in the movie and gaming industries. While recent years have seen

*Equal contribution.

significant progress in 3D object reconstruction for view synthesis using radiance fields [3, 4], these techniques represent an object illuminated by a single fixed environment and they do not enable changing the appearance of the object due to changes in the lighting. This work extends 3D object reconstruction to enable rendering the object under arbitrary illumination.

A typical approach for reconstructing relightable 3D representations from images is inverse rendering: optimizing the material and lighting parameters that together explain the captured images. This is particularly challenging for real-world captures as it is brittle and sensitive to mismatches between the real world's physical light transport and the simplified lighting and material models used during optimization. Furthermore, due to the problem's inherent ambiguities, object properties recovered by inverse rendering often appear implausible and unrealistic when viewed under novel lighting.

At the same time, recent relighting diffusion models [5, 6, 7] have demonstrated impressive capabilities for generating realistic images of objects under arbitrary illumination. However, they only generate a single relit image at a time, which results in inconsistent relighting results when applied to a sequence of viewpoints. While these inconsistently-relit samples can be reconstructed into a single 3D representation [7], optimizing a new 3D representation for each new target lighting is tedious and precludes interactive use cases.

We propose a strategy for distilling samples from a relighting diffusion model into a relightable 3D Neural Radiance Field (NeRF) that can be rendered from arbitrary viewpoints under arbitrary novel environment illumination. Given images of an object, we first use a multi-view diffusion network to generate view-consistent relit images under a wide array of environment illuminations. We then use these multi-view multi-illumination samples to train a novel NeRF architecture that predicts outgoing view-dependent color conditioned on a target illumination.

We evaluate the efficiacy of our method on the task of 3D relighting using both synthetic and real-world benchmarks. Our approach achieves state-of-the-art results but also demonstrates significantly improved test-time performance compared to prior work. These performance gains are due to our generalizable feed-forward relightable NeRF.

2 Related Work

Inverse Rendering for Relighting. Recovering relightable 3D representations of objects from images is a longstanding goal in computer vision and graphics. The prevalent approach is inverse rendering: decomposing the object's appearance into its underlying geometry, illumination, and material parameters, and relighting the object by simulating the physical interaction of a new target illumination with the recovered geometry and materials [8, 9, 10, 11].

Modern methods for reconstructing relightable 3D representations with inverse rendering use differentiable rendering techniques [12, 13] to optimize mesh [14, 15], distance field [16], or volumetric [17, 18, 19, 20, 21, 22, 23, 24] representations of object geometry and material parameters. While these inverse rendering approaches can be effective, they come with significant limitations: errors in estimating an object's geometry and materials can produce unrealistic appearance when the object is relit under a new illumination and physically-accurate relighting involves computationally-expensive Monte Carlo simulation of global illumination, which can be too slow for interactive use cases.

Precomputed Radiance Transfer and Light Stages. Early approaches for interactive relighting in the field of computer graphics proposed the idea of Precomputed Radiance Transfer (PRT) [25]. As appearance behaves linearly with respect to lighting, an object or scene's appearance can be precomputed under a set of basis lightings and these can be linearly combined to produce appearance under any desired target lighting condition. While PRT-based techniques enable interactive relighting, they struggle with the memory demands of storing the precomputed transfer matrices. To enable rendering under novel illuminations from arbitrary viewpoints, PRT-based methods need to store a full transfer matrix for each 3D point in the scene. These memory requirements are prohibitive even for modestly sized scenes and environment maps, so a large body of work focuses on compressing these PRT matrices [26, 27].

One-light-at-a-time (OLAT) captures in light stages [28] can be thought of as directly capturing basis vectors of a real object's radiance transfer matrices. Since each image captured by the light stage is of the object illuminated by a single element of a standard lighting basis, they can be linearly combined to reproduce the object's appearance under any desired environment illumination.

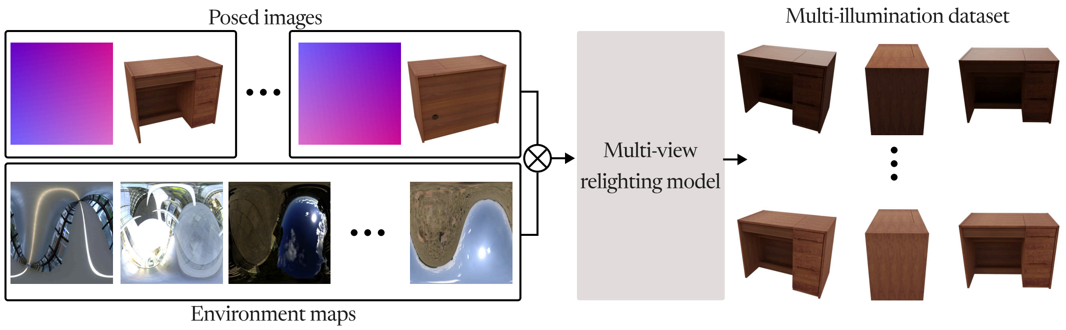


Figure 2: **Multi-view Relight Diffusion**. Our multi-view relighting diffusion model takes as input N posed images illuminated with a consistent, but unknown illumination, represented by camera raymaps and the source pixel values, and an environment map per image that has been rotated to the camera pose. The diffusion model generates images of the same object from the same poses, but lit by an input environment map. To generate our multi-illumination dataset, we repeat this relighting process M times with M environment maps.

Direct Relighting without Inverse Rendering. In the deep learning era, many methods have trained networks to directly output relit images. Early techniques utilized standard convolutional neural networks [29, 30] and more recent approaches are based on powerful diffusion models [5, 7, 6, 31]. These generative relighting models have produced impressive results for image relighting, but they cannot be easily used for 3D relighting as the relit appearance is often not consistent across views. IllumiNeRF [7], MIS [31], and Neural Gaffer [6] perform 3D relighting by reconciling samples from a 2D image relighting diffusion model into a single NeRF. However, the relit 3D representation needs to be optimized separately for each novel target illumination. Recent work on 3D reconstruction from images of an object with varying illumination [32] uses a multi-view diffusion model to relight them to have consistent illumination. However, this approach does not allow relighting using a new unseen illumination condition, so it does not allow for generalization. And it does not utilize the high-frequency details in environment maps. Zeng et al. [33] and RNG [34] utilize shadow and highlight cues as conditioning signals for NeRF. In contrast, we explicitly use the encoded features of incident light from the specular reflection direction. Additionally, in contrast to all of these approaches, our proposed method trains a generalizable relightable NeRF that can be rendered under any target illumination without additional optimization. A concurrent work, RelitLRM [35], directly recovers 3D geometry and appearance but does not guarantee consistent geometry across environment maps. In contrast, our method reconstructs constant geometry under varying illumination. DiffusionRenderer [36] employs a video diffusion model for relighting but lacks 3D consistency, while our approach enforces it by explicitly modeling geometry and rendering. Moreover, our model enables fast inference, avoiding the long sampling times typical of diffusion-based generation.

3 Method

Given a set of N posed images $\mathcal{D} = \{(I_i, \pi_i)\}_{i=1}^{N}$ of an object, where I_i is the i-th image and π_i its pose (including camera extrinsic and intrinsic parameters), we are interested in learning the parameters θ of a relighting function $\mathbf{f}_\theta$, that allows rendering the object from any viewpoint π illuminated by any lighting E (e.g. an environment map) to produce a new image $I = \mathbf{f}_\theta(E, \pi)$. In order to learn the parameters θ of this transformation, we propose to generate a dataset of pairs of posed images with their corresponding illumination to train $\mathbf{f}_\theta$ in a supervised manner. In Sec. 3.1 we will describe how to generate such paired data using a generative relighting diffusion model, and in Sec. 3.2 we explain how to optimize $\mathbf{f}_\theta$, which we implement using a lighting-conditioned, relightable radiance field.

3.1 Multi-View Diffusion Relighting

Our goal is to train a generative multi-view diffusion model $\mathbf{g}$ to relight our given posed images $\mathcal{D}$, which are jointly captured under the same unknown lighting. The diffusion model provides samples from the distribution of possible images $\mathcal{D}_E$ relit by the target illumination E:

$$p(\mathcal{D}_E | \mathcal{D}, E). \tag{1}$$

Our work crucially relies on a multi-view diffusion relighting model to provide consistent relit images, which can then be distilled into a relightable 3D model. This is in contrast with prior work on relighting using diffusion models [7, 6] which independently relights single images and results in ambiguities that must be resolved at the 3D reconstruction stage.

Our network architecture is inspired by multi-view diffusion architectures such as CAT3D [37], which start with pretrained 2D image diffusion models and inflate them by adding cross-attention layers to process multiple views. We adapt such a scheme for the task of relighting and show our architecture in Fig. 2. Since we use a latent diffusion model (LDM), we first map all images and environment maps into the latent space of the original 2D diffusion model. In particular, the environment map is encoded in a similar manner to Neural Gaffer [6]: We use two separate latents corresponding to high dynamic range (HDR) and low dynamic range (LDR). This representation captures both bright details like direct light sources, as well as relatively dim sources like diffuse objects. Like Neural Gaffer, we also use standard tone mapping for the LDR environment map [38, 39], and for the HDR encoding we apply logarithmic tone mapping followed by normalization to $[0, 1]$ by subtracting the minimal value and dividing by the maximal value. Additionally, for each image I_i in our dataset, we apply a 3D rotation to the environment map to align it with the corresponding camera pose π_i. We combine the HDR and LDR encodings of the environment map with the encoded image and the ray map of the pose of each image by concatenating them, and then apply self-attention. Details of the network architecture for our multi-view relighting diffusion models are provided in Fig. 1 of the supplementary material.

3.2 Relightable Neural Radiance Field

Our diffusion model $\mathbf{g}$ generates consistent relit images given a target lighting environment. Our end goal is to obtain a 3D representation of the object that can modify the illumination of the scene *without* additional per-illumination optimization. To do this, we first use the multi-view relighting model to create a new dataset $\mathcal{D}_{\text{relit}}$ for each object by taking the N images in the original dataset $\mathcal{D}$ and relighting them using a collection of M environment maps $\mathcal{E} = (E_1, ..., E_M)$. The new relit dataset of an object can be written as:

$$\mathcal{D}_{\text{relit}} = \{\mathbf{g}(I_i, \pi_i, E_j) \colon i = 1, ..., N,\ j = 1, ..., M\}, \tag{2}$$

where $\mathbf{g}(I_i, \pi_i, E_j)$ is the diffusion model's estimate for the ith image I_i (whose pose is π_i) lit by the jth environment map E_j. $\mathcal{D}_{\text{relit}}$ contains $N \times M$ images.

We then use the multi-illumination dataset $\mathcal{D}_{\text{relit}}$ to train a NeRF. Since we wish to fit the NeRF model to our dataset with varying lighting so that it generalizes to novel lighting environments *outside* of our training set of illuminants $\mathcal{E}$, care must be taken when designing the model's architecture. We use NeRF-Casting [40] as the base NeRF model since it efficiently captures view-dependent appearance, and we modify it to allow for conditioning on the illumination. Crucially, we encode the environment maps using two types of conditioning signals: general conditioning and specular conditioning. The general conditioning encodes the entire environment map into an embedding that is fed to the appearance MLP and is designed as a general-purpose, low-frequency signal for relighting, while the specular conditioning only encodes incident light coming from the specular direction. Its goal is to improve the model's capacity to capture high-frequency reflections (similar to prior work on reconstructing specular reflections in NeRF [41]). The conditioning signals are illustrated in Fig. 3.

In order to encode the specular appearance of a camera ray, NeRF-Casting [40] casts a small set of reflected rays, traces them through the NeRF's geometry, and volume renders a feature vector $\bar{\mathbf{f}}$ that encodes the scene content observed by these reflected rays. In our setting, we provide the two conditioning signals by concatenating them to the feature vector $\bar{\mathbf{f}}$ of each ray, which is mapped by NeRF-Casting's MLP to the ray's specular color component.

General Conditioning. Our *general conditioning* signal encodes the full environment map, and is therefore a complete description of the lighting of the object. To do this, we train an transformer-based encoder which maps the environment map into a vector. While the idea of using a per-lighting encoding is similar to the approach of NeRF-in-the-Wild [42], a crucial difference is that our embeddings are not optimized to fit to individual images, but they are instead parameterized as a learned mapping from the environment maps. This enables us to render the scene under novel illumination at inference time without requiring any additional training.

To encode the illumination features from the environment map, we utilize a transformer encoder with self-attention. Our transformer is trained from scratch, and its architecture is based on ViT-S8 of DINO [43], followed by a single matrix multiplication W to produce a compact 128-dimensional embedding vector:

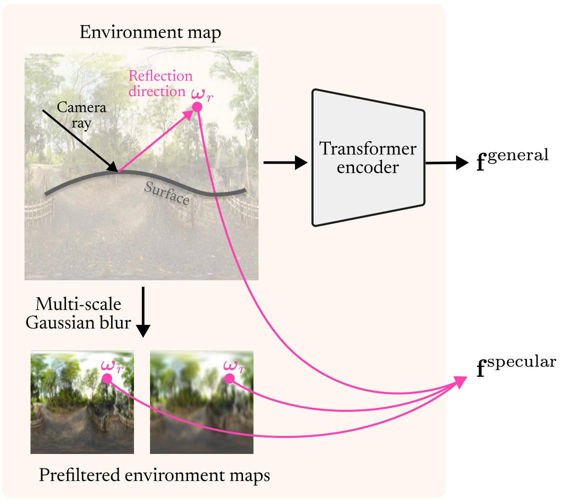
$$\mathbf{f}^{\text{general}} = W \cdot \text{ViT}(E) \tag{3}$$

Specular Conditioning. Although the general conditioning vector theoretically contains all information necessary for relighting, we found it to be insufficient for reconstructing and rendering high-frequency specular highlights. Our *specular conditioning* is designed to fix that by explicitly encoding incident light from the specular reflection direction, similar to the encoding used in prior work for reconstructing reflective objects and scenes [41, 40, 44]. Instead of only sampling the environment map value at the reflection direction, we use a series of blur kernels centered around the reflection direction to simulate the effects of materials with different roughnesses. The i-th component of this conditioning vector can be written as:

$$\mathbf{f}_i^{\text{specular}} = \int_{\mathbb{S}^2} E(\boldsymbol{\omega}')\mathrm{G}(\boldsymbol{\omega}'; \boldsymbol{\omega}_r, \sigma_i)d\boldsymbol{\omega}' \tag{4}$$

where $\mathrm{G}(\boldsymbol{\omega}'; \boldsymbol{\omega}_r, \sigma_i)$ is a Gaussian blur kernel around $\boldsymbol{\omega}_r$ with width σ_i, and $\boldsymbol{\omega}_r$ is the view direction reflected about the surface normal. For efficiency, we preprocess the environment map by blurring it at all directions using Eq. 4, and then query it at the reflection direction $\boldsymbol{\omega}_r$ during the NeRF optimization stage.

Network Predictions. Following NeRF-Casting [40], our relightable NeRF takes as input the 3D coordinates, ray direction, general conditioning, and specular conditioning. The geometry MLP predicts density, roughness, normals, and geometry features, while the color MLP outputs the RGB values.



Environment map
Camera ray
Reflection direction ω_r
Surface
Transformer encoder
$\mathbf{f}^{\text{general}}$
Multi-scale Gaussian blur
ω_r
ω_r
$\mathbf{f}^{\text{specular}}$
Prefiltered environment maps

Figure 3: **Lighting conditioning signals**. We use a combination of two lighting conditioning signals to train the NeRF on our generated multi-illumination dataset. The general lighting encoding $\mathbf{f}^{\text{general}}$ is used for encoding the full environment map in a single embedding, and is obtained using a transformer encoder applied to the entire sphere of incident radiance. The specular encoding $\mathbf{f}^{\text{specular}}$ is composed of the environment map value, as well as prefiltered versions of the environment map, queried at the reflection direction $\boldsymbol{\omega}_r$, which is the direction of the camera ray reflected about the surface normal vector. Combining these two conditioning signals provides the NeRF with all the information necessary for relighting diffuse materials as well as shiny ones, which exhibit strong reflections.

4 Experimental Setup

4.1 Datasets

Training datasets. To train multi-view relighting diffusion, we use a dataset of 400k synthetic 3D objects, including 100k from Objaverse [46]. Each object is rendered in 64 views $\times$ 16 HDR illuminations, with environments sampled from Polyhaven [47] (590 maps) and augmented via random up-axis rotations.

Evaluation datasets. For relighting evaluations, we used two datasets: TensoIR [45] and Stanford-ORB [48]. TensoIR is a synthetic benchmark, which contains renderings of four objects under six lighting conditions. We use the train split of 100 views with "sunset" lighting condition as inputs for relightable NeRF. We then evaluate 200 novel views under other five environment maps, including "bridge", "fireplace", "forest", "city", and "night". In total, we have 4,000 renderings for evaluation metric calculation. Stanford-ORB is real-world benchmarks by data capture in the wild. It has 14 objects composed of various materials. Each object is captured under three distinct lighting conditions, producing a total of 42 (object, lighting) combinations. Following its evaluation protocol,

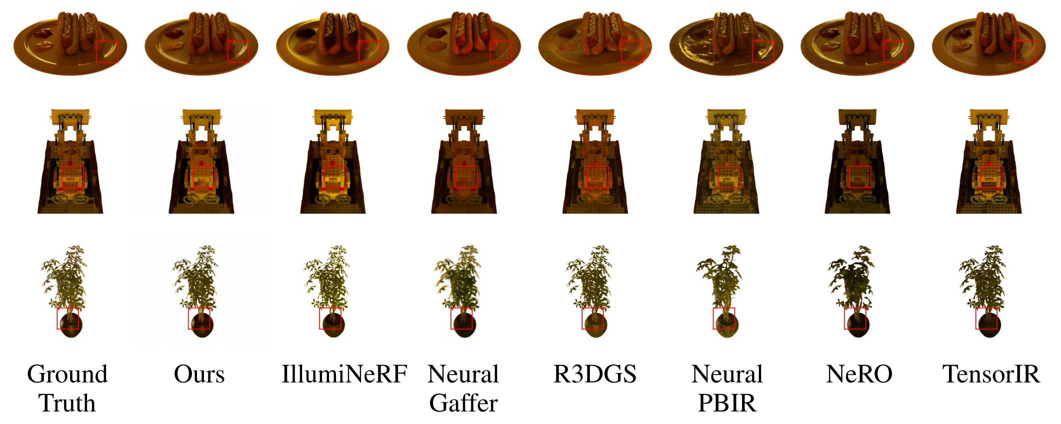


Figure 4: **Qualitative comparisons on TensoIR [45].** All renderings are rescaled to the image resolution of the ground truth. Compared to previous works, our method recovers more plausible specular highlights and more accurate colors as indicated with the red box.

we use images of an object under a single lighting condition and evaluate novel views under the two target lighting settings.

4.2 Implementation details

Multi-View Diffusion model. We fine-tune our model starting from a pre-trained latent image generation model, as described in [49]. The multi-view denoiser is derived from the CAT3D network architecture [37], with modifications to the input channel dimensions to align with our relighting task. The inputs to our model are images with resolution 512×512 which are encoded to $64 \times 64 \times 8$ latents. We chose the number of views to be 64. The model was trained on 128 TPU v5 chips using a learning rate of 10^{-4}, with a total batch size of 128 for 360k iterations. After training, we generate the multi-illumination dataset by running our relighting diffusion inference on 111 environment maps.

Relightable NeRF model. We train our NeRF on 8 H100s for 500k steps. We use a 512×512 resolution environment map as the target illumination. We sample each reflection rays 3 times; once using a point sample on the full resolution environment map, and then using Gaussian kernels of sizes 20×20 and 40×40 pixels in radius with σ_i values of 10 and 20 respectively (see Fig. 3). In order to maximize the number of Illumination conditions we use for training, we make several reductions to the size of model relative to the NeRF-Casting architecture. We lower the batch size to 1,000 and increase the number of training steps to 500,000. We also decrease the size of the "bottleneck" vector $\mathbf{b}$ in both the geometry and appearance MLPs relative to NeRF-Casting. Please refer to supplementary material for more details on the base architecture.

4.3 Baselines

We compare our method against existing inverse rendering methods including NeRFactor [50], InvRender [51], PhySG [52], NeRD [19], NVDiffRecMC [14], NVDiffRec [15], Neural-PBIR [53], NeRO [44],TensoIR [45], and recent single-view relighting diffusion methods IllumiNeRF [7] and Neural Gaffer [6]. We also include the most recent Gaussian splatting-based inverse rendering method R3DG [54].

4.4 Evaluation Metrics

We evaluate the relighting rendering quality by PSNR, SSIM [55], and LPIPS-VGG [56].

5 Results

Our method is the top-performing technique on existing relighting benchmarks for synthetic and real-world objects. Furthermore, it can render images from novel viewpoints under novel illuminations at

Method	PSNR-H ↑	PSNR-L ↑	SSIM ↑	LPIPS ↓
NVDiffRecMC [57] †	25.08	32.28	0.974	0.027
NVDiffRec [15] †	24.93	32.42	0.975	0.027
PhySG [52]	21.81	28.11	0.960	0.055
NVDiffRec [15]	22.91	29.72	0.963	0.039
NeRD [19]	23.29	29.65	0.957	0.059
NeRFactor [50]	23.54	30.38	0.969	0.048
InvRender [51]	23.76	30.83	0.970	0.046
NVDiffRecMC [57]	24.43	31.60	0.972	0.036
Neural-PBIR [53]	26.01	33.26	0.979	0.023
R3DG [54]	21.25	27.50	0.962	0.063
Neural Gaffer [6]	23.16	29.94	0.966	0.047
IllumiNeRF [7]	25.56	32.74	0.976	0.027
Ours	26.21	32.91	0.980	0.027

Table 2: **StandfordORB benchmark [48].** We evaluate fourteen objects captured in the real world. Each object was captured in three different lightings. For each object-lighting pair, we evaluate novel view renderings under the other two lighting. The benchmark contains 840 renderings in total. † denotes models trained with the ground-truth 3D scans and pseudo materials optimized from light-box captures. Best and 2nd-best are highlighted.

interactive speeds (0.384 seconds per frame). The only other methods that achieves similar relighting and rendering speeds are ones based on Gaussian splatting combined with inverse rendering, such as R3DG (0.415 seconds per frame), but the quality of these methods is significantly lower than of generative methods or inverse rendering methods that are based on NeRF.

5.1 TensoIR benchmark

In Fig. 4, our method achieves state-of-the-art performance on the TensoIR benchmark, improving upon the ability of prior works to capture specularities in the reflective "hot dog" and "ficus" scenes while accurately capturing diffuse appearance from global illumination in the "lego" and "armadillo" scenes. The superiority of our method can also be verified by quantitative results in Tab. 1.

Method	PSNR ↑	SSIM ↑	LPIPS ↓
NeRFactor [50]	23.38	0.908	0.131
InvRender [51]	23.97	0.901	0.101
TensoIR [45]	28.58	0.944	0.081
Neural-PBIR [53]	27.09	0.925	0.085
NeRO [44]	27.00	0.935	0.074
R3DG [54]	29.05	0.937	0.080
NeuralGaffer [6]	27.30	0.918	0.122
IllumiNeRF [7]	29.71	0.947	0.072
Ours	30.74	0.950	0.069

Table 1: **TensoIR benchmark [45].** We evaluate all four objects in the benchmark, each under five novel lightings. Each object is rendered from 200 views for novel view evaluation under each lighting. Thus, we have 4,000 renderings in total for quantitative evaluation. Best and 2nd-best are highlighted.

5.2 Stanford-ORB benchmark

In Fig. 5, our method also achieves state-of-the-art performance on the Stanford-ORB dataset, and is most effective at reflective objects like the "baking" and "ball" scenes where consistent reflections clearly show up in the reconstruction. We provide quantitative comparisons in Tab. 2, where our method outperforms others in PSNR-H and SSIM, and acheive second-best results in PSNR-L and LPIPS. As discussed in IllumiNeRF [7], our results are also qualitatively superior to those of Neural-PBIR [53], but they are worse quantitatively due to the mostly-diffuse renderings of Neural-PBIR.

5.3 Real-world Objects

Finally, we demonstrate the ability of our method to relight "in-the-wild" captures of real objects with spatially-varying material properties under natural lighting in Fig. 1. Since we have no ground truth images for real-world relighting evaluation, we only show qualitative results. Our method captures convincing specularities on the wood and metal parts of the model sewing machine, as well as accurate shadows cast by the arm of the sewing machine.

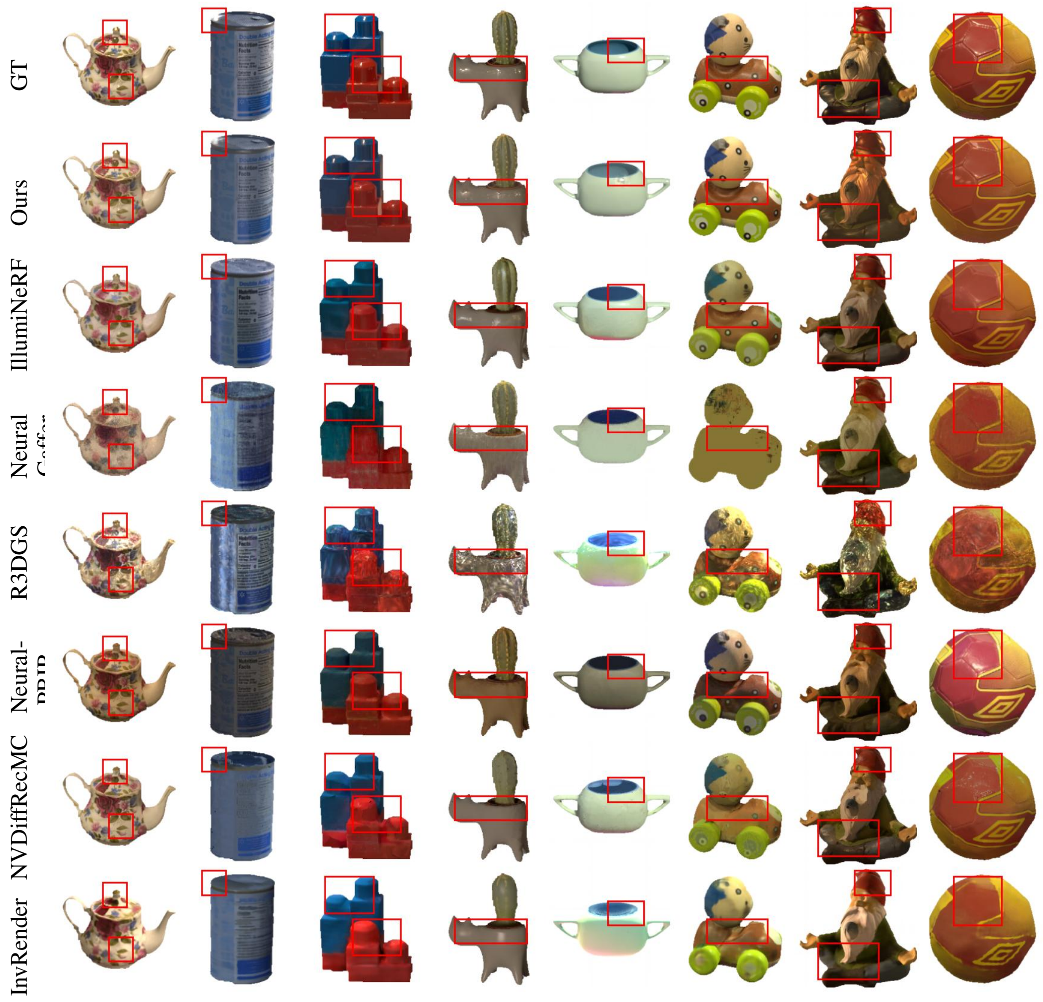


Figure 5: **Qualitative comparisons on Stanford-ORB [48].** Renderings from all methods are rescaled to the image resolution of the ground truth. Compared to previous work, our method produces high-fidelity renderings with more faithful specular reflections highlighted in the red boxes.

Method	PSNR ↑	SSIM ↑	LPIPS ↓
(a) No blurring in specular conditioning	31.35	0.90	0.080
(b) No specular conditioning	30.00	0.88	0.079
(c) No general conditioning	21.59	0.70	0.130
(d) Per-image appearance embeddings	19.12	0.62	0.160
(e) 128 × 128 environment map	29.26	0.89	0.082
(f) 64 × 64 environment map	27.69	0.86	0.110
Our full model, 64-view, 111 envmaps	31.88	0.91	0.075

Table 3: **Ablation studies on the “hotdog” scene from TensoIR [45].** See the text and Fig. 6 for corresponding qualitative results and additional explanations. Best and 2nd-best are highlighted.

5.4 Ablation Studies

We next perform an ablation study of the different components of our method, done on the “hotdog” scene of the TensoIR benchmark, chosen since it has the most interesting materials in that dataset. Each ablation is reported in a row of Tab. 3 and its corresponding column in Fig. 6.

Multi-scale specular conditioning. Our multi-scale specular conditioning features, which are computed by blurring the environment map using multiple kernel sizes as in Eq. 4, are provided to the model. Each scale is designed to approximate the incident light averaged over a set of directions corresponding to a particular surface roughness. When we skip this blurring operation and use the

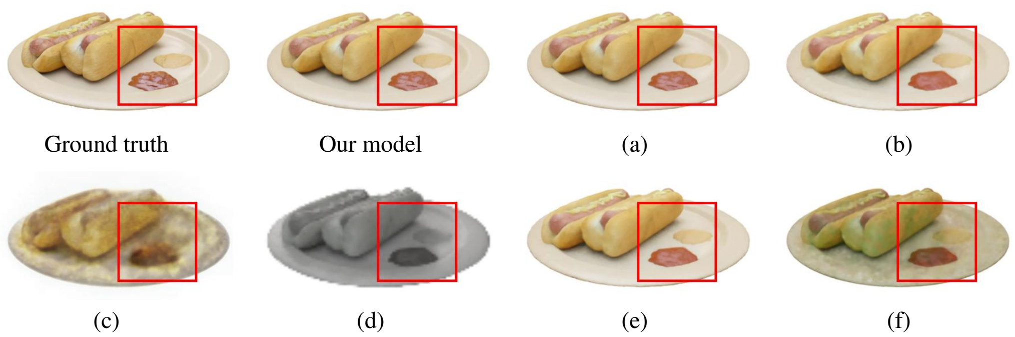


Figure 6: **Qualitative ablation studies.** We compare a ground truth image and the relighting results of our model with a set of ablations (a)-(f). (a) Using specular conditioning without blurring results in high-quality images with slightly lower metrics, whereas (b) removing the specular conditioning altogether greatly reduces the accuracy of specular highlights. (c) We observe that our model produces significant artifacts when the general conditioning is removed or (d) replaced with per-image appearance embeddings; and that providing environment maps at lower resolutions of (e) 128×128 or (f) 64×64 tends to blur specular highlights and introduce rendering artifacts.

Method	PSNR ↑	SSIM ↑	LPIPS ↓
Ours full model, 4 view, 111 envmaps	31.04	0.86	0.082
Ours full model, 16 view, 111 envmaps	31.86	0.90	0.077
Our full model, 64-view, 10 envmaps	29.12	0.82	0.086
Our full model, 64-view, 50 envmaps	31.78	0.88	0.079
Our full model, 64-view, 111 envmaps	31.88	0.91	0.075
Our full model, 64-view, 150 envmaps	31.90	0.92	0.075

Table 4: **Ablation studies of number of views and illulimantions on the "hotdog" scene from TensoIR [45].** Best and 2nd-best are highlighted.

environment map values directly, our model can still represent highly specular regions (see Fig. 6(a)), but struggles more with rough or diffuse surfaces, leading to a drop in reconstruction metrics.

Specular conditioning. Next we remove the specular conditioning signal altogether, which leads to another small drop in reconstruction metrics as well as a qualitative drop in the accuracy of rendered specular highlights.

General conditioning. Removing the general conditioning signal from our network results in significant artifacts and poor reconstruction metrics, since the general conditioning is the main mechanism for providing the target illumination to the relighting model.

Per-image appearance embeddings. An alternative to our conditioning signals is a per-image appearance embedding vector, similar to the GLO codes in NeRF-in-the-Wild [42]. While this allows the model to be trained on multiple illuminations, it does not generalize to new unseen lights. Additionally, unlike our model, we found that the embedding vectors do not scale well to a large number of illumination conditions, resulting in significantly worse qualitative and quantitative results.

Environment map resolution. Our full model uses conditioning signals based on an environment map of resolution 512×512. Computing the conditioning signals from environment maps of size 128×128 results in loss of detail in the specular highlights. Further lowering the resolution to 64×64 results in rendering artifacts even for diffuse surfaces.

Number of views. Our full model learns the joint distribution of 64-view relighting. To analyze the impact of the number of views in the diffusion model, we compare relighting novel view synthesis results using 4, 16, and 64-view diffusion on the hotdog scene of TensoIR dataset. As shown in Tab. 4, the 64-view diffusion model consistently outperforms the others across all metrics. This indicates that jointly denoising more views leads to more consistent and higher-quality relit images.

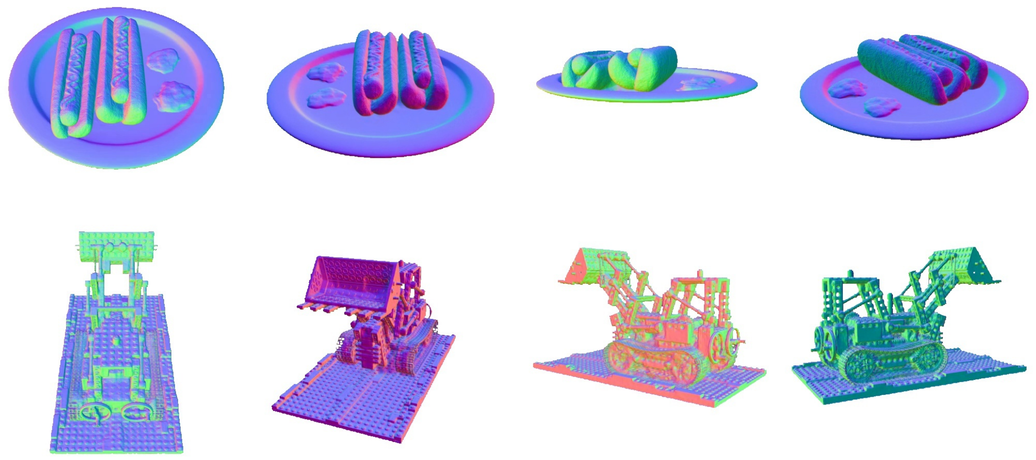


Figure 7: **Normal map rendering of our relightable NeRF on objects from the TensoIR dataset.**

Number of environment maps. We also study the effect of environment map count on relightable NeRF training by using 10, 50, 111, and 150 illuminations. As shown in Tab. 4, more illuminations generally improve relighting performance, which saturates around 111 maps.

Normal map visualization. As shown in Fig. 7, our relightable neural radiance fields can render high-quality normal maps, which are comparable to prior inverse rendering techniques like TensoIR, as well as novel view synthesis approaches like NeRF-Casting which explicitly encourage geometry to be more surface-like.

5.5 Limitations

While our method improves 3D object relighting by achieving more accurate specularities and fast inference, it can still be further improved in several aspects. First, although our relighting diffusion model is trained on objects with materials varying in diffuse albedo and roughness, which are the main sources of variation in real-life materials, we did not train on objects exhibiting phenomena such as subsurface scattering, refraction, or volumetric effects. Expanding our training data to include these complex materials would improve robustness and generalizability to real data. Second, we use environment maps to define lighting conditions, which assumes that the light sources are infinitely far away from the object. Exploring more general lighting models containing near-field illumination components could enhance realism in diverse illumination scenarios. Finally, our approach focuses on object-centric scenes, and extending it to large-scale scene relighting would be an exciting direction for future research.

6 Conclusion

This paper introduces a novel method for 3D object relighting, enabling fast, feed-forward relighting during inference. By modeling a virtual light stage with a generative multi-diffusion model, we create a diverse dataset of multi-illumination images. This dataset serves as a prior to train a light-conditioned Neural Radiance Field (NeRF) model, which subsequently learns to render the object under arbitrary target illumination conditions. Existing methods for 3D relighting often rely on inverse rendering techniques, constrained by shading models limited to specific material types, or they directly generate relit NeRFs, embedding the lighting within the model itself. This requires retraining the NeRF for each new lighting condition. In contrast, our proposed model exhibits generalization across diverse lighting conditions at inference, facilitating efficient feed-forward relighting. Experimental results demonstrate the effectiveness of our method in relighting complex real-world objects with high fidelity.

Acknowledgement. We would like to thank Xiaoming Zhao, Rundi Wu, Songyou Peng, Ruiqi Gao, Stan Szymanowicz, Hadi Alzayer, Ben Poole, Aleksander Holynski, Jason Zhang, Jonathan T. Barron, Peter Hedman, Alex Trevithick, and Jiahui Lei for their valuable contributions. We also extend our gratitude to Shlomi Fruchter, Kevin Murphy, Mohammad Babaeizadeh, Han Zhang and Amir Hertz for training the base text-to-image latent diffusion model.

References

[1] Paul Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Acm siggraph 2008 classes*, pages 1–10. 2008.

[2] Paul Debevec, Tim Hawkins, Chris Tchou, Haarm-Pieter Duiker, Westley Sarokin, and Mark Sagar. Acquiring the reflectance field of a human face. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 145–156, 2000.

[3] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

[4] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D gaussian splatting for real-time radiance field rendering. 2023.

[5] Chong Zeng, Yue Dong, Pieter Peers, Youkang Kong, Hongzhi Wu, and Xin Tong. Dilightnet: Fine-grained lighting control for diffusion-based image generation. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–12, 2024.

[6] Haian Jin, Yuan Li, Fujun Luan, Yuanbo Xiangli, Sai Bi, Kai Zhang, Zexiang Xu, Jin Sun, and Noah Snavely. Neural gaffer: Relighting any object via diffusion. In *Advances in Neural Information Processing Systems*, 2024.

[7] Xiaoming Zhao, Pratul P Srinivasan, Dor Verbin, Keunhong Park, Ricardo Martin Brualla, and Philipp Henzler. Illuminerf: 3d relighting without inverse rendering. *arXiv preprint arXiv:2406.06527*, 2024.

[8] Ravi Ramamoorthi and Pat Hanrahan. A signal processing framework for inverse rendering. In *SIGGRAPH*, 2001.

[9] Yizhou Yu, Paul Debevec, Jitendra Malik, and Tim Hawkins. Inverse global illumination: Recovering reflectance models of real scenes from photographs. In *SIGGRAPH*, 1999.

[10] Yoichi Sato, Mark D Wheeler, and Katsushi Ikeuchi. Object shape and reflectance modeling from observation. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 379–387, 1997.

[11] Hendrik PA Lensch, Jan Kautz, Michael Goesele, Wolfgang Heidrich, and Hans-Peter Seidel. Image-based reconstruction of spatial appearance and geometric detail. *ACM Transactions on Graphics (TOG)*, 22(2):234–257, 2003.

[12] Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. Differentiable monte carlo ray tracing through edge sampling. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, 37(6):222:1–222:11, 2018.

[13] Merlin Nimier-David, Delio Vicini, Tizian Zeltner, and Wenzel Jakob. Mitsuba 2: A retargetable forward and inverse renderer. *Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 38(6), December 2019.

[14] Jon Hasselgren, Nikolai Hofmann, and Jacob Munkberg. Shape, Light, and Material Decomposition from Images using Monte Carlo Rendering and Denoising. *arXiv:2206.03380*, 2022.

[15] Jacob Munkberg, Jon Hasselgren, Tianchang Shen, Jun Gao, Wenzheng Chen, Alex Evans, Thomas Müller, and Sanja Fidler. Extracting triangular 3d models, materials, and lighting from images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8280–8290, 2022.

[16] Kai Zhang, Fujun Luan, Zhengqi Li, and Noah Snavely. Iron: Inverse rendering by optimizing neural sdfs and materials from photometric images. In *CVPR*, 2022.

[17] Sai Bi, Zexiang Xu, Pratul P. Srinivasan, Ben Mildenhall, Kalyan Sunkavalli, MiloÅą HaÅąan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Neural reflectance fields for appearance acquisition. In *arXiv.*, 2020.

[18] Pratul P. Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T. Barron. NeRV: Neural reflectance and visibility fields for relighting and view synthesis. In *CVPR*, 2021.

[19] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. Nerd: Neural reflectance decomposition from image collections. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12684–12694, 2021.

[20] Xiuming Zhang, Pratul P. Srinivasan, Boyang Deng, Paul Debevec, William T. Freeman, and Jonathan T. Barron. NeRFactor: Neural factorization of shape and reflectance under an unknown illumination. In *SIGGRAPH Asia*, 2021.

[21] Alexander Mai, Dor Verbin, Falko Kuester, and Sara Fridovich-Keil. Neural microfacet fields for inverse rendering. In *ICCV*, 2023.

[22] Haian Jin, Isabella Liu, Peijia Xu, Xiaoshuai Zhang, Songfang Han, Sai Bi, Xiaowei Zhou, Zexiang Xu, and Hao Su. Tensoir: Tensorial inverse rendering. In *CVPR*, 2023.

[23] Benjamin Attal, Dor Verbin, Ben Mildenhall, Peter Hedman, Jonathan T Barron, Matthew OâĂŹToole, and Pratul P Srinivasan. Flash cache: Reducing bias in radiance cache based inverse rendering. In *European Conference on Computer Vision*, pages 20–36. Springer, 2024.

[24] Xiang Feng, Chang Yu, Zoubin Bi, Yintong Shang, Feng Gao, Hongzhi Wu, Kun Zhou, Chenfanfu Jiang, and Yin Yang. Arm: Appearance reconstruction model for relightable 3d generation. *arXiv preprint arXiv:2411.10825*, 2024.

[25] Peter-Pike Sloan, Jan Kautz, and John Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *SIGGRAPH*, 2002.

[26] Ren Ng, Ravi Ramamoorthi, and Pat Hanrahan. All-frequency shadows using non-linear wavelet lighting approximation. In *SIGGRAPH*, 2003.

[27] Peter-Pike Sloan, Jesse Hall, John Hart, and John Snyder. Clustered principal components for precomputed radiance transfer. In *SIGGRAPH*, 2003.

[28] Paul Debevec, Tim Hawkins, Chris Tchou, Haarm-Pieter Duiker, Westley Sarokin, and Mark Sagar. Acquiring the reflectance field of a human face. In *SIGGRAPH*, 2000.

[29] Zexiang Xu, Kalyan Sunkavalli, Sunil Hadap, and Ravi Ramamoorthi. Deep image-based relighting from optimal sparse samples. *ACM Transactions on Graphics (TOG)*, 37(4):126, 2018.

[30] Tiancheng Sun, Jonathan T Barron, Yun-Ta Tsai, Zexiang Xu, Xueming Yu, Graham Fyffe, Christoph Rhemann, Jay Busch, Paul E Debevec, and Ravi Ramamoorthi. Single image portrait relighting. *ACM Transactions on Graphics (TOG)*, 38(4):79–1, 2019.

[31] Yohan Poirier-Ginter, Alban Gauthier, Julien Phillip, J-F Lalonde, and George Drettakis. A diffusion approach to radiance field relighting using multi-illumination synthesis. In *Computer Graphics Forum*, volume 43, page e15147. Wiley Online Library, 2024.

[32] Hadi Alzayer, Philipp Henzler, Jonathan T Barron, Jia-Bin Huang, Pratul P Srinivasan, and Dor Verbin. Generative multiview relighting for 3D reconstruction under extreme illumination variation. *arXiv preprint arXiv:2412.15211*, 2024.

[33] Chong Zeng, Guojun Chen, Yue Dong, Pieter Peers, Hongzhi Wu, and Xin Tong. Relighting neural radiance fields with shadow and highlight hints. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–11, 2023.

[34] Jiahui Fan, Fujun Luan, Jian Yang, Miloš Hašan, and Beibei Wang. Rng: Relightable neural gaussians. *arXiv preprint arXiv:2409.19702*, 2024.

[35] Tianyuan Zhang, Zhengfei Kuang, Haian Jin, Zexiang Xu, Sai Bi, Hao Tan, He Zhang, Yiwei Hu, Milos Hasan, William T Freeman, et al. Relitlrm: Generative relightable radiance for large reconstruction models. *arXiv preprint arXiv:2410.06231*, 2024.

[36] Ruofan Liang, Zan Gojcic, Huan Ling, Jacob Munkberg, Jon Hasselgren, Chih-Hao Lin, Jun Gao, Alexander Keller, Nandita Vijaykumar, Sanja Fidler, et al. Diffusion renderer: Neural inverse and forward rendering with video diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 26069–26080, 2025.

[37] Ruiqi Gao, Aleksander Holynski, Philipp Henzler, Arthur Brussee, Ricardo Martin-Brualla, Pratul P. Srinivasan, Jonathan T. Barron, and Ben Poole. Cat3d: Create anything in 3d with multi-view diffusion models. *Advances in Neural Information Processing Systems*, 2024.

[38] Samuel W Hasinoff, Dillon Sharlet, Ryan Geiss, Andrew Adams, Jonathan T Barron, Florian Kainz, Jiawen Chen, and Marc Levoy. Burst photography for high dynamic range and low-light imaging on mobile cameras. *ACM Transactions on Graphics (ToG)*, 35(6):1–12, 2016.

[39] Paul Debevec, Erik Reinhard, Greg Ward, and Sumanta Pattanaik. High dynamic range imaging. In *ACM SIGGRAPH 2004 Course Notes*, pages 14–es. 2004.

[40] Dor Verbin, Pratul P Srinivasan, Peter Hedman, Ben Mildenhall, Benjamin Attal, Richard Szeliski, and Jonathan T Barron. Nerf-casting: Improved view-dependent appearance with consistent reflections. In *SIGGRAPH Asia 2024 Conference Papers*, pages 1–10, 2024.

[41] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. Ref-NeRF: Structured view-dependent appearance for neural radiance fields. *CVPR*, 2022.

[42] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In *CVPR*, 2021.

[43] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

[44] Yuan Liu, Peng Wang, Cheng Lin, Xiaoxiao Long, Jiepeng Wang, Lingjie Liu, Taku Komura, and Wenping Wang. Nero: Neural geometry and brdf reconstruction of reflective objects from multiview images. 2023.

[45] Haian Jin, Isabella Liu, Peijia Xu, Xiaoshuai Zhang, Songfang Han, Sai Bi, Xiaowei Zhou, Zexiang Xu, and Hao Su. Tensoir: Tensorial inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2023.

[46] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13142–13153, 2023.

[47] Greg Zaal, Rob Tuytel, Rico Cilliers, James Ray Cock, Andreas Mischok, Sergej Majboroda, Dimitrios Savva, and Jurita Burger. Polyhaven: a curated public asset library for visual effects artists and game designers, 2021.

[48] Zhengfei Kuang, Yunzhi Zhang, Hong-Xing Yu, Samir Agarwala, Elliott Wu, Jiajun Wu, et al. Stanford-orb: a real-world 3d object inverse rendering benchmark. *Advances in Neural Information Processing Systems*, 36, 2024.

[49] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[50] Xiuming Zhang, Pratul P Srinivasan, Boyang Deng, Paul Debevec, William T Freeman, and Jonathan T Barron. Nerfactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics (ToG)*, 40(6):1–18, 2021.

[51] Yuanqing Zhang, Jiaming Sun, Xingyi He, Huan Fu, Rongfei Jia, and Xiaowei Zhou. Modeling indirect illumination for inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18643–18652, 2022.

[52] Kai Zhang, Fujun Luan, Qianqian Wang, Kavita Bala, and Noah Snavely. PhySG: Inverse rendering with spherical gaussians for physics-based material editing and relighting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5453–5462, 2021.

[53] Cheng Sun, Guangyan Cai, Zhengqin Li, Kai Yan, Cheng Zhang, Carl Marshall, Jia-Bin Huang, Shuang Zhao, and Zhao Dong. Neural-pbir: reconstruction of shape, material, and illumination. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18046–18056, 2023.

[54] Jian Gao, Chun Gu, Youtian Lin, Zhihao Li, Hao Zhu, Xun Cao, Li Zhang, and Yao Yao. Relightable 3d gaussians: Realistic point cloud relighting with brdf decomposition and ray tracing. In *European Conference on Computer Vision*, pages 73–89. Springer, 2025.

[55] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[56] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[57] Jon Hasselgren, Nikolai Hofmann, and Jacob Munkberg. Shape, light, and material decomposition from images using monte carlo rendering and denoising. *Advances in Neural Information Processing Systems*, 35:22856–22869, 2022.

[58] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, et al. Jax: composable transformations of python+ numpy programs. 2018.

[59] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[60] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[61] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[62] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. In *European Conference on Computer Vision*, pages 1–18. Springer, 2024.

[63] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016.

[64] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4015–4026, 2023.

Appendix – ROGR: Relightable 3D Objects using Generative Relighting

A Supplementary Webpage and Videos

We suggest readers check the website in our supplementary material. We provide more video renderings, including ablation studies and result comparisons on TensoIR [22], StanfordORB [48], and real-world objects.

B Additional Implementation Details

B.1 Multi-view Relighting Diffusion

We implement our multi-view relighting diffusion model using JAX [58]. It is initialized from a pre-trained latent diffusion model for text-to-image generation, similar to StableDiffusion [49]. Our model denoises multiple noisy latents of size $64 \times 64 \times 8$ and decodes them into multi-view relit images of size $512 \times 512 \times 3$. Since our model is not conditioned on text prompt, we only feed empty strings to the CLIP text encoder [59].

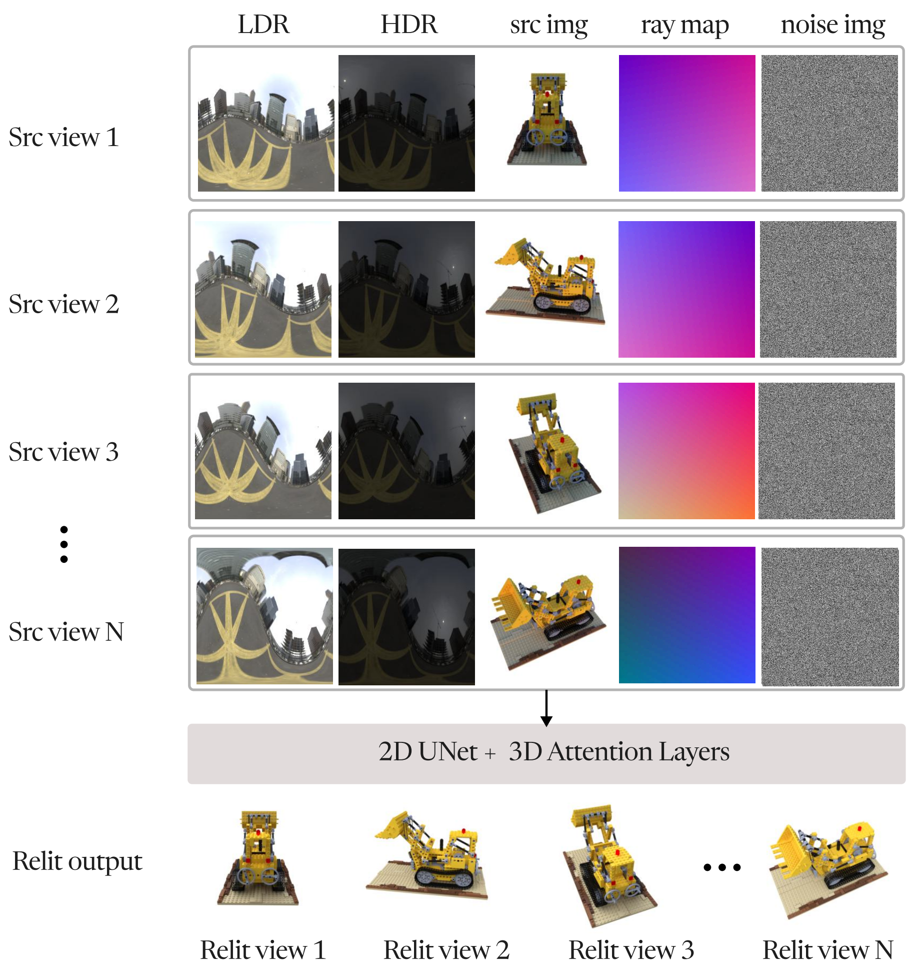


Figure 8: **Multi-view Relighiting Diffusion Models.** For each view, we concatenate noisy image latents, raymaps containing pose information, source image latents, HDR and LDR environment latents as inputs, and feed them into a multi-view denoiser network that is implemented by 2D UNet with additional 3D Attention layers. 2D UNet individually processes the latent feature from each view. 3D Attention layers flatten multi-view image latents into a 1D sequence, and then perform self-attention to exchange information across different pixels and views.

In Fig. 8, we provide the detailed inputs of our relighting diffusion model. While our diffusion model receives and produces image latents, we depict them as images for better clarity. We encode RGB images under source lighting, HDR, and LDR target environment maps into latents of size $64 \times 64 \times 8$. The raymaps consist of ray origins and ray directions corresponding to image pixels. We downscale them from the original image resolution $512 \times 512 \times 6$ to the latent size $64 \times 64 \times 6$. We concatenate the source image latent, HDR, LDR environment latents, raymaps, and noisy latents along the channel dimension to form a new feature of size $64 \times 64 \times 38$, and then feed it into a multi-view diffusion network to produce clean target latents of size $64 \times 64 \times 8$. Our denoiser network is based on CAT3D [37]. Please refer to Fig. 7 of CAT3D for the network architecture details.

During training, we use the DDPM schedule, with beta values that linearly increase from 8.5×10^{-4} to 1.2×10^{-2} over 1024 steps. We use noise prediction as our diffusion objective. The model was trained on 128 TPU v5 chips using a learning rate of 10^{-4}, with a total batch size of 128 for 360k iterations and 10K warm-up steps. We adopted a progressive training scheme, where we first trained a 4-view diffusion model for 300k steps, and then fine-tuned it for 16-view diffusion for 15k steps, and finally fine-tuned it for 64-view diffusion for 45 steps. We keep the learning rate as 10^{-4} when we fine-tune the model to relight the large number of views. We enable classifier-free guidance (CFG) [60] by randomly dropping the HDR and LDR environment maps with a probability of 0.1. During inference, we use the DDIM schedule [61] with 50 sampling steps and the classifier-free weight is set to 3.0.

B.2 Relightable NeRF

Our relightable NeRF is mainly based on NeRF-Casting [40], which implicitly learnt the accumulated reflection features to model accurate and detailed reflections. Instead, we choose to explicitly learn these from given enviroment maps. Given a single ray with origin $\mathbf{o}$ and direction $\mathbf{d}$, we sample N points $\mathbf{x}^{(i)}$ along the ray and use multi-resolution hash grid and MLP to encode $\mathbf{x}^{(i)}$ into density $\tau^{(i)}$, roughness $\rho^{(i)}$, and surface normal $\mathbf{n}^{(i)}$. Then, they are alpha composited to compute a single expected termination point $\overline{\mathbf{x}}$, a von Mishes-Fisher distribution (vMF) width $\overline{k}$, and surface normal $\overline{\mathbf{n}}$. Next, we construct a reflection cone by reflecting $\mathbf{d}$ around the micro-surface to obtain a vMF distribution over reflected rays $vMF(\mathbf{d}', \overline{k})$. We sample $K = 5$ reflected rays with location $\mathbf{o}'$ and $\mathbf{d}'_j$. We cast these rays and sample N' points $\mathbf{x}_j^{(i)}$ along each reflection ray. N' points $\mathbf{x}_j^{(i)}$ are encoded into N' densities. Based on location $\mathbf{o}'$ and $\mathbf{d}'_j$, we can query illumination information from environment maps and encode them with $\mathbf{x}_j^{(i)}$ into features $\mathbf{f}_j^{(i)}$ through an MLP. These features $\mathbf{f}_j^{(i)}$ are alpha-composed along each ray to get per-ray reflection feature $\overline{\mathbf{f}}^{(i)}{}_j$, which can be further averaged into a single reflection feature $\mathbf{f}$. Finally, $\mathbf{f}$, bottleneck geometry feature $\mathbf{b}^{(i)}$, mixing coefficients $\beta^{(i)}$ and view direction $\mathbf{d}$ are feed into color decoder to predict the color value of $\mathbf{x}^{(i)}$.

C Data

C.1 Training data preprocessing

We use Objaverse [46] and an internal dataset containing high-quality 3D assets as our training data. The Objaverse dataset is released under the Apache-2.0 license.

To ensure high-quality renderings for training, we filter out low-quality 3D assets from Objaverse using the object list provided in [62], and further exclude (semi-)transparent objects, as our focus is on reflective and shiny surfaces. For 3D assets lacking texture or material information, we assign a uniform color sampled from $[0, 1]$ as texture. We randomly sample three values from $[0, 1]$ as the diffuse, roughness, and metallic terms of the material model.

Our relighting diffusion model requires multi-view images under diverse lighting conditions. To this end, we use 590 equirectangular environment maps from [47]. For each object, we randomly sample 64 camera poses on a sphere centered around the object. The camera distance ranges from $[0.5, 2.0]$. For each camera view, we randomly select 16 environment maps, augmenting each with a random horizontal shift. Then we use Blender's Cycles path tracer to render images at a resolution of 512×512, with 512 samples per pixel.

C.2 Evaluation benchmarks.

We use TensoIR [22] and StanfordORB [48] datasets for relighting evaluations. TensoIR is under the MIT license. StanfordORB does not specify a license in the official GitHub repository.

C.3 Real-world data capture

We capture real-world objects to evaluate our relighting model using a Sony DSLR camera. Each object is placed on a table, and a handheld camera is used to record an image sequence by moving around the object. Each sequence contains approximately 160 to 300 frames. Camera intrinsics and extrinsics are estimated using COLMAP[63] for NeRF training. For foreground segmentation, we apply the pre-trained SAM [64] model. We will release our captured dataset upon publication under a non-commercial academic license.

C.4 Evaluation metrics

We evaluate the relighting rendering quality by PSNR, SSIM [55], and LPIPS-VGG [56] for low dynamic range (LDR) images. On the Stanford-ORB benchmark, we also compute PSNR for high dynamic range (HDR) images, denoted as PSNR-H, while PSNR for LDR images is referred to as PSNR-L. For methods that only produce LDR images, we apply the inverse process of the sRGB tone mapping to transform the outputs into linear values. To address ambiguities in the relighting task, we align the predicted outputs with the ground truth images by applying channel-wise scale factors prior to metric computation. For Stanford-ORB, these scale factors are determined separately for each output image. For TensoIR, a single global scale factor is calculated and uniformly applied to all output images.

D Social Impacts

Our work presents an algorithm for photorealistic object relighting, which can provide an immersive experience for VR/AR products. It has the potential to simplify labor-intensive workflows in 3D content creation and empower artists in the shopping, film, and gaming industries. It can also be used to augment 3D/multi-view datasets with diverse lighting conditions, potentially benefiting downstream tasks that rely on large-scale photorealistic 3D/multi-view training data. However, this technology also carries potential risks. Misuse of the relighting capability might enable the creation of fraudulent or harmful visual content. Additionally, our diffusion models require significant computational resources for training, which could bring environmental concerns due to high electricity consumption.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: See abstract and Sec. 1.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See Sec. 5.5 of the supplementary material.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: No theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: See Sec. 4 and Sec. B of the supplementary material, and video results from our website.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We have not cleaned and released our data and code.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so âĂIJNoâĂİ is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: See Sec. 4 and Sec. B of the supplementary material.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We did not report error bars or statistical significance in our experiments due to the high computational cost of our model.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See the Sec. 4 and Sec. B of the supplementary material.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We reviewer the NeurIPS Code of Ethics and preserve anonymity.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: See the Sec. D of the supplementary material.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [No]

    Justification: We have not cleaned and released our data and code.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: See Sec. C of the supplementary material.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The LLM is used only for writing, editing, or formatting purposes in our paper.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.