Neural Factorization of Shape, Material, and Lighting under Unknown Illumination

Burak Çuhadar
Technical University of Munich
burak.cuhadar@tum.de

Mohamed Ebbed
Technical University of Munich
mohamed.ebbed@tum.de

Yue Chen
Technical University of Munich
yue.chen@tum.de

Abstract

We propose a method for factorizing scenes into geometry, material, and lighting from a given RGB Images with known camera poses. We represent the geometry by explicit 3D Mesh converted from neural Signed Distance Function (SDF) with Marching Cubes algorithm [6]. We use Mitsuba2 [10] path tracer to determine the surface intersection points and the visibility of each light source in the scene w.r.t. each intersection. We use MLPs to predict BRDF and environment illumination. Particularly, the BRDF module contains one MLP for albedo estimation, and one MLP for BRDF latent code prediction. A pretrained latent space, which is pretrained with MERL [7] dataset, takes the BRDF latent code and predicts the material specularity. The MLPs are optimized with one single re-rendering loss. This factorization enables free-viewpoint relighting and material editing. Qualitative and quantitative experiments show that our model outperforms previous methods and achieves state-of-the-art results.

1. Introduction

Factorizing a scene into geometry, material and lighting from given input images with known poses has been a challenging problem in computer graphics. Such factorization for a scene facilitates re-rendering the scene while varying its illumination and material properties. In this work we propose a solution for decomposing scenes under one unknown illumination. The key idea is to leverage the advantages of implicit SDF geometry representation and explicit mesh representation. The implicit SDFs are ideal for the geometry reconstruction due of the ease of back-propagation, while the explicit meshes have huge advantages of accuracy and efficiency in surface intersection point calculation. We first reconstruct the geometry with pretrained SDF from NeuS [14], then transform the SDF into a mesh by Marching Cube algorithm. After that, we can analyse surface intersection points and light visibility of the light sources in the scene, which have a huge benefits for rendering proper cast shadows. The material properties are represented as Bidirectional Reflectance Distribution Function (BRDF), which is optimized with MLPs and a pretrained BRDF prior learned from real world data. We applied a novel neural environment map for estimating the lighting. Our model is optimized using one single re-rendering loss and smoothness constrains. We obtain the state-of-the-art results qualitatively and quantitatively. Our pipeline is visualized in Figure 1.

To summarize, our main contribution are

- A method that performs neural factorization for scenes given multi-view images with known camera poses under one unknown illumination.
- Performing detailed 3D geometry reconstruction with implicit SDF and physic-based surface intersections and light visibility estimation with explicit mesh.
- Reducing the computation time for the visibility estimation from 150+ hours (our baseline) to 5 minutes.
- Presenting a novel upsampling technique during the inference that improves the relighting results without additional training.
- Achieving the state-of-the-art results.

2. Related Work

Novel View Synthesis In the recent few years, utilizing neural fields to represent 3D scenes has gained a lot of attention. In NeRF [8], the scene is represented by MLPs that

map 3D coordinates and viewing directions to volume density and radiance. The view is rendered by sampling points on each camera ray to the scene and feeding these points with viewing direction to the MLPs which are optimized to minimize the re-rendering loss between the ground truth image and the rendered image which is rendered by volume rendering. Another similar approach to NeRF is NeuS [14] which represents the geometry as neural SDFs and uses a novel sphere tracing to locate the surface. Those methods are not able to factorize the scene into explicit components like material or lighting. In our work, we can decompose the scene represented by neural SDFs and ensures a plausible relighting and material editing.

Neural Factorization and Inverse Rendering Despite that neural fields have amazing performance in novel view synthesis, it is not possible to factorize the represented scenes as the neural fields only represents specific properties like volume density, radiance and SDF. A work that builds on Nerf to solve this problem is NeRFactor [15] which factorizes scenes represented by NeRF. The scene geometry and normals are extracted from NeRF's density volume which is noisy in contrary to the Neural SDFs that we use. The other properties are represented by MLPs that are optimized by re-rendering loss in addition to optimizing the 2D environment map that represents scene illumination. In our work we represent the light using a neural representation. Other similar work NeRD [3] optimizes the neural fields for novel view synthesis and factorization simultaneously. Our approach is different than their work as our representation for BRDF is neural representation while they use analytical BRDF. As the problem of factorization is highly unconstrained, other works tried to have some assumptions to make the optimization easier. For example NeRV [13] has assumed multiple known illumination, while in our work we assume one unknown illumination.

Normalizing Flows For physically based rendering, we need to model a BRDF that conserves energy which means that for every incoming light direction integration of BRDF over the hemisphere of reflected light directions should be less than or equal to one. This constraint is not straightforward to satisfy with neural BRDF approaches. To achieve that, Chen et al. [4] proposes to use the idea of Normalizing Flows [11] to train an MLP representing BRDF where the integration mentioned above satisfies the energy conservation property. In our work, we decided to implement a conditional variant of Real NVP [5] similar to the work done by Ardizzone et al. [1]. To condition Real NVP, we input condition variables to the scale and shift MLPs.

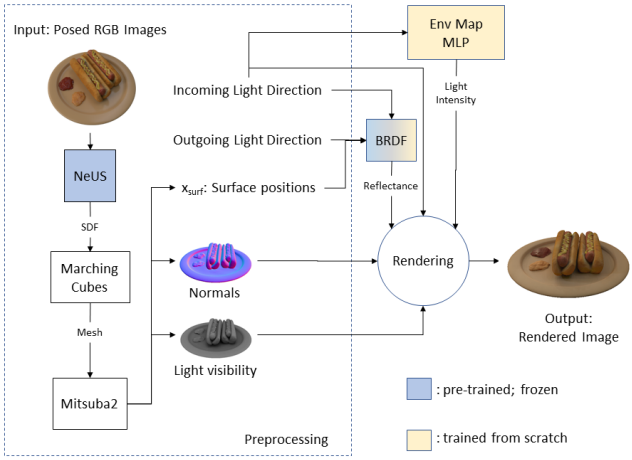


Figure 1. An overview of our pipeline. Given input RGB images with known camera poses. We apply NeuS [14] as our geometry reconstruction and extract a 3D Mesh from the trained neural SDF with marching cubes. Then Light visibility, surface positions and normals are calculated using Mitsuba2 [10] path tracer. Then MLPs are queried to predict reflectance, albedo and light intensity. Then the scene is rendered with the rendering equation

3. Method

3.1. Assumption

In this section we will state the assumptions that we have on the scenes that our method factorizes.

- We assume that the scene consists of only hard surfaces and non-transparent materials. For example, materials like glass and smoke won't be counted.
- We assume a single unknown spatial invariant illumination.
- We assume single-bounce reflections. Our model considers only direct illumination.

3.2. Shape

We first applied NeuS [14] to reconstruct the scene geometry. NeuS takes multi-view images and camera poses as input and provides a high-quality geometry reconstruction as implicit SDF. We then transform the SDF into explicit mesh using Marching Cube algorithm. A mesh representation enables an efficient and accurate estimation of surface intersection points and light visibility.

Surface Intersection Points and Normals In order to calculate the surface positions x_{surf} and the normals $n_{x_{surf}}$ at these positions of the scene, we apply Mitsuba2 path tracer to shoot rays from the camera to the 3D mesh and calculate their first intersection. We adopt the normal of the

mesh where the intersection is found as the surface normal of the intersection point.

Light Visibility After determining surface positions, we use Mitsuba2 to calculate light visibility $l_v(x_{surf}, \omega_{in})$ for every surface intersections and uniformly sampled incoming light directions ω_{in}. The visibility is a binary value that determines whether a ray shooting from the light source in ω_{in} direction has intersection with the mesh.

3.3. Material

We represent BRDF by decomposing it into diffuse and non-diffuse parts following the approach used in NeRFactor by Zhang et al. [15]. They model BRDF at each surface location $\boldsymbol{x}_{\text{surf}}$, representing albedo and specular component separately:

$$\boldsymbol{R}(\boldsymbol{x}_{\text{surf}}, \boldsymbol{\omega}_{\text{i}}, \boldsymbol{\omega}_{\text{o}}) = \frac{\boldsymbol{a}(\boldsymbol{x}_{\text{surf}})}{\pi} + \boldsymbol{f}_{\text{r}}\left(\boldsymbol{x}_{\text{surf}}, \boldsymbol{\omega}_{\text{i}}, \boldsymbol{\omega}_{\text{o}}\right), \quad (1)$$

where $\boldsymbol{a}$ is albedo and $\boldsymbol{f}_{\text{r}}$ is spatially-varying specular BRDF. The incoming and outgoing directions are denoted by $\boldsymbol{\omega}_{\text{i}}$ and $\boldsymbol{\omega}_{\text{o}}$, respectively. Our whole BRDF pipeline is depicted in Figure 2.

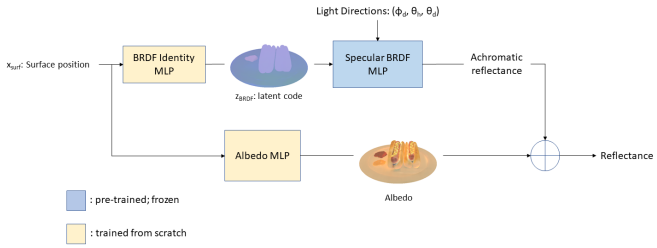


Figure 2. Our BRDF pipeline: Our specular BRDF MLP is pre-trained on the MERL [7] dataset to learn a latent space of real materials. During joint optimization, we optimize latent codes using our BRDF Identity MLP. For the diffuse component of BRDF we optimize an Albedo MLP. To get the final reflectance, we sum up the outputs from Albedo MLP with specular BRDF MLP.

Albedo The albedo $\boldsymbol{a}$ is represented as an MLP $\boldsymbol{f}_{\text{a}} : \boldsymbol{x}_{\text{surf}} \mapsto \boldsymbol{a}$. For each surface location $\boldsymbol{x}_{\text{surf}}$, this MLP estimates the albedo. It is optimized jointly with other components using re-rendering loss and smoothness prior which is formulated as:

$$\ell_{\text{a}} = \lambda_a \sum_{\boldsymbol{x}_{\text{surf}}} \frac{1}{3} \left\| \boldsymbol{f}_{\text{a}}(\boldsymbol{x}_{\text{surf}}) - \boldsymbol{f}_{\text{a}}(\boldsymbol{x}_{\text{surf}} + \boldsymbol{\epsilon}) \right\|_1, \quad (2)$$

where λ_a is a hyperparameter set to 0.05 and $\boldsymbol{\epsilon}$ is a random 3D perturbation from $\boldsymbol{x}_{\text{surf}}$ sampled from a zero-mean Gaussian with 0.01 standard deviation.

Specularity The specular part of BRDF is first pre-trained on a real-world BRDF dataset called MERL [7] to learn a latent space of real-world materials. The MERL dataset provides RGB reflectance values for isotropic materials. Therefore the incoming and outgoing directions are parameterized by Rusinkiewicz coordinates [12] with 3 degrees of freedom $(\phi_{\text{d}}, \theta_{\text{h}}, \theta_{\text{d}})$. We denote the latent code with $\boldsymbol{z}_{\text{BRDF}}$ which is 3-dimensional. During pre-training, we train an MLP whose inputs are the latent code concatenated with the Rusinkiewicz coordinates, that outputs an achromatic reflectance $\boldsymbol{r}$:

$$\boldsymbol{f}'_{\text{r}} : \left(\boldsymbol{z}_{\text{BRDF}}, (\phi_{\text{d}}, \theta_{\text{h}}, \theta_{\text{d}})\right) \mapsto \boldsymbol{r}, \quad (3)$$

where $\boldsymbol{f}'_{\text{r}}$ denotes our re-parameterization of $\boldsymbol{f}_{\text{r}}$. We optimize both the latent codes and the weights of this MLP during pre-training. The loss is log mean squared error between the prediction and the ground-truth achromatic reflectance values obtained from MERL dataset. The latent codes are initialized with zero-mean Gaussian with a standard deviation of 0.01.

During the joint optimization, this MLP is frozen and $\boldsymbol{z}_{\text{BRDF}}$ are predicted using another MLP which we call BRDF identity MLP:

$$\boldsymbol{f}_{\text{z}} : \boldsymbol{x}_{\text{surf}} \mapsto \boldsymbol{z}_{\text{BRDF}} \quad (4)$$

The BRDF identity MLP is optimized from scratch during the joint optimization using the re-rendering loss and the spatial smoothness loss as in Equation 2:

$$\ell_{\text{z}} = \lambda_z \sum_{\boldsymbol{x}_{\text{surf}}} \frac{\left\| \boldsymbol{f}_{\text{z}}(\boldsymbol{x}_{\text{surf}}) - \boldsymbol{f}_{\text{z}}(\boldsymbol{x}_{\text{surf}} + \boldsymbol{\epsilon}) \right\|_1}{\dim(\boldsymbol{z}_{\text{BRDF}})}, \quad (5)$$

where λ_z is a hyperparameter set to 0.01. Our final BRDF is:

$$\boldsymbol{R}(\boldsymbol{x}_{\text{surf}}, \boldsymbol{\omega}_{\text{i}}, \boldsymbol{\omega}_{\text{o}}) = \frac{\boldsymbol{f}_{\text{a}}(\boldsymbol{x}_{\text{surf}})}{\pi} + \boldsymbol{f}'_{\text{r}}\Big(\boldsymbol{f}_{\text{z}}(\boldsymbol{x}_{\text{surf}}), \phi_{\text{d}}, \theta_{\text{h}}, \theta_{\text{d}}\Big). \quad (6)$$

3.4. Lighting

We represent the incoming light intensity from direction ω_{in} by an Environment Map MLP:

$$f_{\text{l}} : \omega_{in} \mapsto l_{\omega_{in}} \quad (7)$$

where $l_{\omega_{in}}$ is the light intensity from direction ω_{in}.

In our model, we choose 16x32 fixed light incoming direction in order to compare with our baseline which represents the lighting as a latitude-longitude HDR light probe image with a fixed size of 16x32.

3.5. Rendering

Given surface intersections, surface normals, light visibility, material BRDF, and environment lighting, we can now calculate the outgoing radiance at position x_{surf} from the viewing direction ω_o using the rendering equation.

$$
\begin{aligned}
& L_o(x_{surf}, w_o) \qquad (8) \\
&= \int_{\Omega} R(x_{surf}, w_{in}, w_o) f_l(w_{in})(w_{in} \cdot n_{x_{surf}}) dw_{in} \\
&= \sum_{w_i} R(x_{surf}, w_i, w_o) f_l(w_i)(w_i \cdot n_{x_{surf}}) f_v(x_{surf}, w_i) \Delta w_i
\end{aligned}
$$

where the visibility $f_v(x_{surf}, w_i)$ is a binary mask.

The final re-rendering loss ℓ_{rgb} is the mean squared error between the rendered image and the ground truth image. Our full loss is defined as $\ell_{\text{rgb}} + \ell_{\text{a}} + \ell_{\text{z}}$.

4. Experiments

4.1. Experimental Settings

Dataset We evaluate our model with the NeRFactor data set. It contains four synthetic scenes rendered by Blender. Each scene is provided with 200 views for the training, 8 views for the validation, and 100 views for the inference. It provides for each view its ground truth normal, albedo, environment maps and rendered images under multiple illumination. We use only rendered images under one illumination for the training and validation, and use images under other illumination for evaluating the relighting performance.

Implementation Details During the preprocessing, we train NeuS for 500k iterations (512 rays per batch) for 20 hours on a single NVIDIA RTX3070Ti GPU. We transform the NeuS SDF result to mesh with a marching cube resolution of 1024. The calculation of surface intersection points and light visibility takes only several minutes.

We adopt the same architecture of albedo and BRDF z estimation MLPs from NeRFactor, each of which contains four layers, each with 128 hidden units. The neural environment map MLP has a similar architecture but has a value of one added to the MLP output to ensure that the initial illumination values are larger than zero. We use positional encoding for the input coordinates.

In the optimization of the MLPs, we sample 5x1024 shuffled rays every batch from the same camera pose. The albedo smoothness weight is set to 0.05 and the BRDF smoothness weight is set to 0.01. The optimization takes 1.5 hours for 100 epochs on a RTX3070Ti.

4.2. Factorization of Shape, Material, and Lighting

In this experiment, we show how our model factorizes scenes into geometry, diffuse and specular reflectance, and illumination.

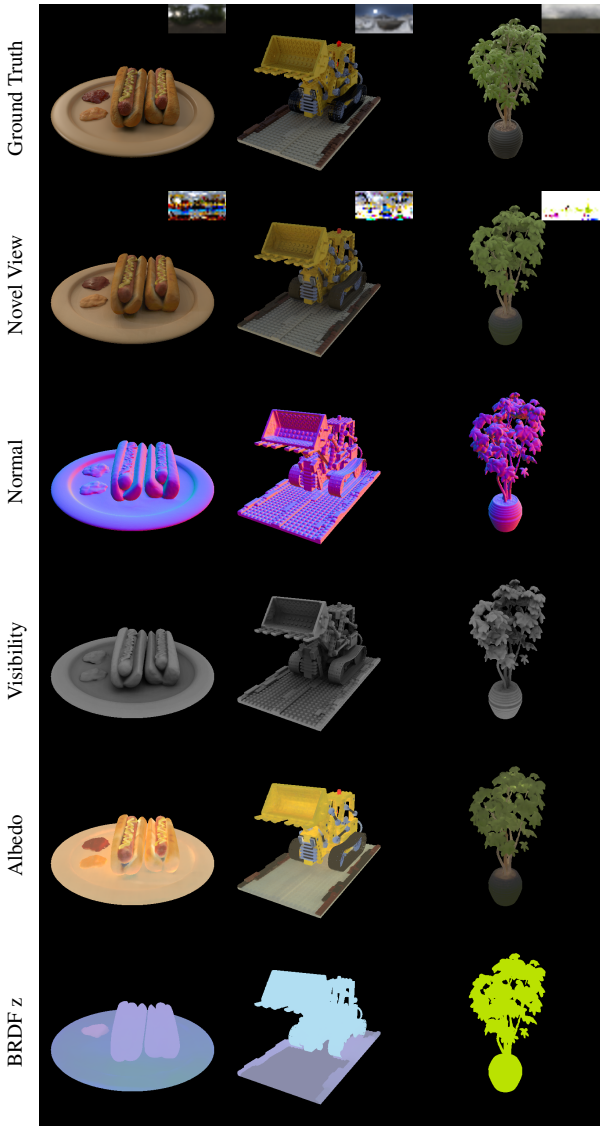


Figure 3. **Factorization of Shape, Material, and Lighting.** Our model performs detailed normal reconstruction, physical-based light visibility analysis, clean albedo estimation with minimal shadows, and reasonable material prediction. Although the estimated environment maps differ from the ground truth, the position of the dominant light sources are determined properly in HDR format.

As shown in Figure 3, our model synthesizes novel view images by factorizing in high-quality normal, reflectance and environment map. Our model reconstructs a precise geometry using NeuS. In the hot dog scene, the roughness of the bun and the smoothness of the plate are accurately recreated. Our model is capible of calculating physic-based light visibility by applying path tracing on explicit 3D mesh. For the material properties, our model manages to analyze distinguished diffuse reflectance (albedo) and reasonable spec-

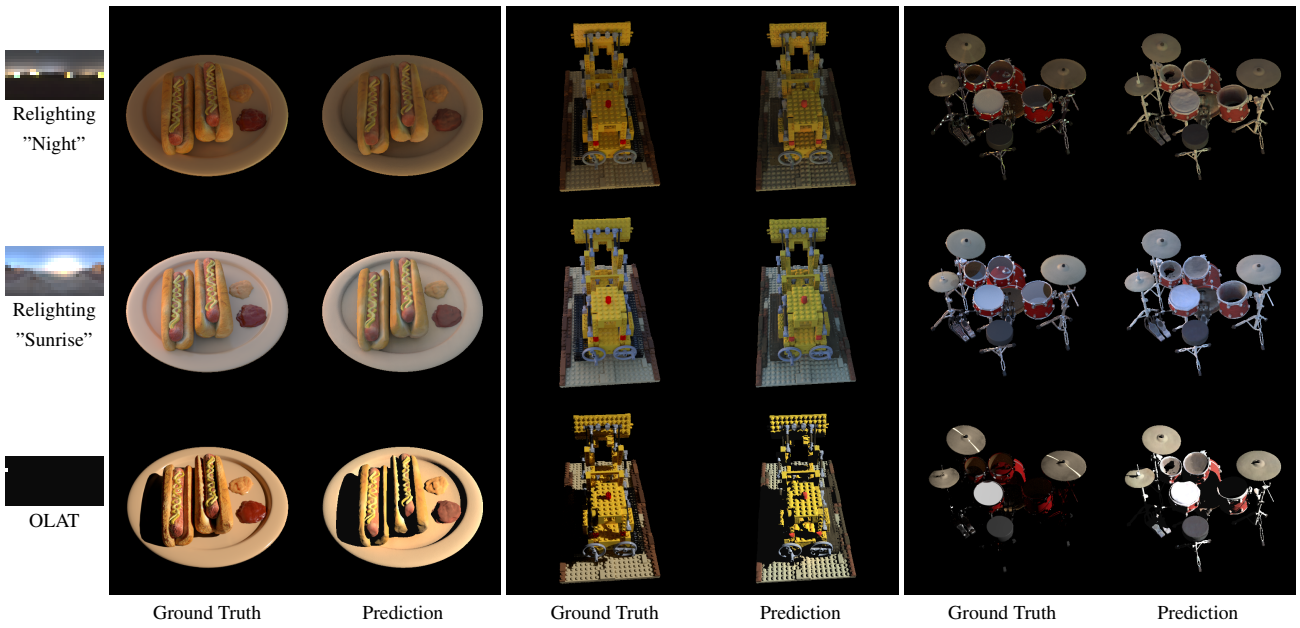


Figure 4. **High-quality Free-viewpoint Relighting.** Our model enables a free-viewpoint image synthesis under arbitrary environment illumination including the challenging OLAT illumination. Our synthesized images closely match to the ground truth with precise colors and realistic cast shadows.

ular reflectance. The specular reflectance is illustrated by the visualization of the latent code (BRDF z), meaning different color indicates different specular effects.

The small images in the upper right corner of Ground Truth or Novel View in Figure 3 represent the ground truth or predicted environment maps. Our model successfully analyzes the position of the dominant light source in the HDR space. It is hard to recognize in our figure because I) we visualize the environment maps in LDR space, therefore the intensity of dominant light sources are truncated, II) our model predicts the non-dominant light sources "brighter" than the ground truth due to the one-bounce reflection assumption and not shiny material, III) objects are only observed from the upper hemisphere so the bottom half of the environment map can not be estimated properly.

Note that the albedo and environment map are calibrated as in previous works [2, 15]. Each RGB channel of the predicted albedo is scaled by a global scalar generated by minimizing the mean squared error between the raw predicted albedo and the ground truth albedo of the first view. Thus, we can disambiguate the relative light intensity of reflectance and environment illumination. The visualization of the environment map is scaled by the same scalar.

4.3. Free-Viewpoint Relighting

Since our model explicitly decomposes the shape, material and lighting of the scene, a free-viewpoint relighting is enabled by replacing the predicted environemnt map with new illumination. Particularly, we test our model under one point light on at a time (OLAT) illumination. The OLAT relighting is an extreme lighting situation that aids in geometry and material estimating artifact identification.

In Figure 4, we demonstrate relighting results under two normal illumination and one OLAT illumination. Our model achieves exceptional performance in all lighting conditions. In hot dog and lego scene, our model predicts accurate color, exact soft shadows under new environment with ambient illumination, and realistic hard shadows with a clean edge under OLAT conditions.

In drums scene, the diffuse color and shadows are estimated correctly, but the transparent and anisotropic reflection are failed to be synthesized. It is anticipated because I) NeuS assumes hard and nontransparent surface, II) our pretrained specularity module is trained with nontransparent isotropic MERL data.

4.4. Comparison

Baseline We compare our model with the state-of-the-art neural factorization model - NeRFactor [15]. NeRFactor extracts surface intersection points and light visibility by accumulating transmittance from a pretrained neural radiance Field (NeRF) [8]. The surface normal is calculated by taking the derivative of NeRF's σ-volume w.r.t. x. The light visibility and surface normal are further smoothed with MLPs. For the lighting estimation, NeRFactor adopts an HDR light probes with a fix size of 16x32.

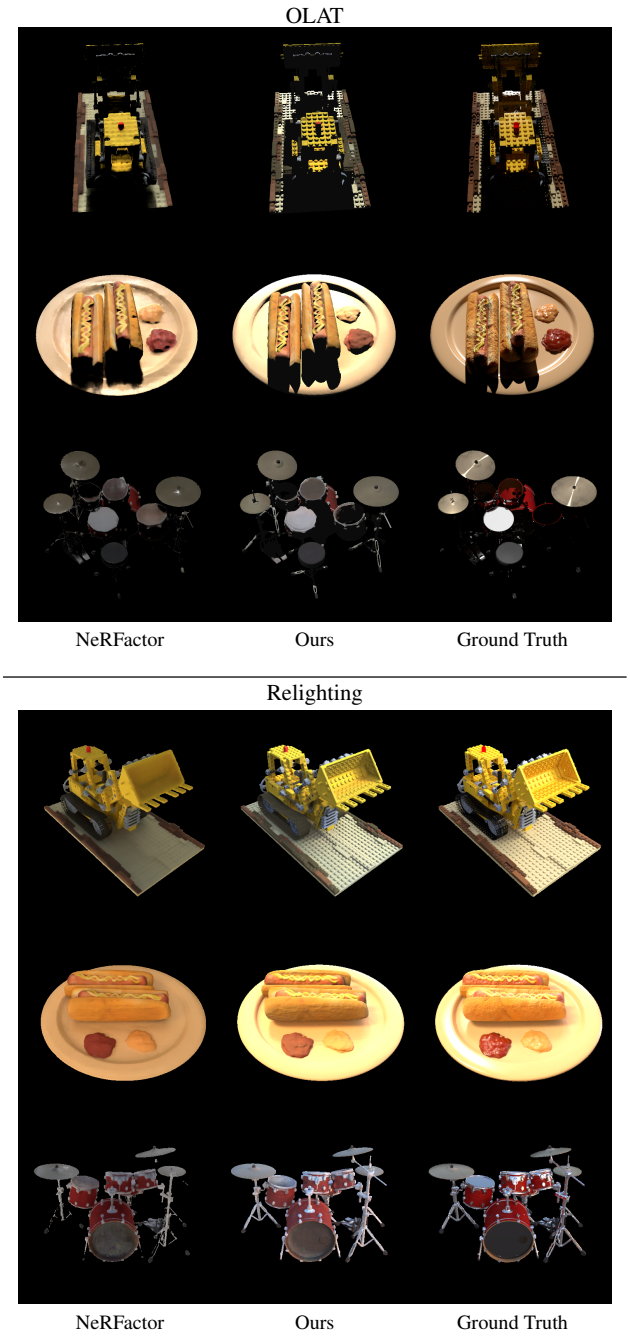


Figure 5. **Comparison against NeRFactor [15].** Our model can estimate more detailed geometry, more realistic hard shadows with sharp boundaries, more reasonable soft shadows, and more precise color and brightness in relighting.

Relighting Results Comparison Table 1 shows a quantitative comparison of albedo estimation, novel view synthesis, and relighting results. Our model outperforms NeRFactor in all aspects.

A qualitative comparison of relighting results is shown in Figure 5. We can observe that the details in our synthetic images are of outstanding grade. In lego scene, we can identify each stud in our prediction, however in NeRFactor, the studs are oversmoothed. Besides, our model can better anticipate shadows. Under OLAT conditions, we can observe realistic hard shadows with clear boundaries, but NeRFactor's shadows have artifacts and fuzzy edges. Finally, our relighting results are closer to the ground truth with a more precise color and brightness.

	Albedo			**Novel View Synthesis**			**Relighting**		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
NeRFactor	22.273	0.907	0.106	21.217	0.867	0.118	21.794	0.852	0.119
Ours	23.771	0.914	0.099	26.874	0.905	0.074	23.366	0.884	0.087

Table 1. **Quantitative Comparison.** The metrics are the mean value (each value is calculated between one predicted image and the corresponding ground truth pairwise) of all four synthetic scenes (hotdog, ficus, lego, and drums) over 200 test views. We relight the scenes under 8 novel lighting conditions (without OLAT, since the data set does not provide enough ground truth OLAT results for test split). The background color of NeRFactor results and the ground truths are set to black.

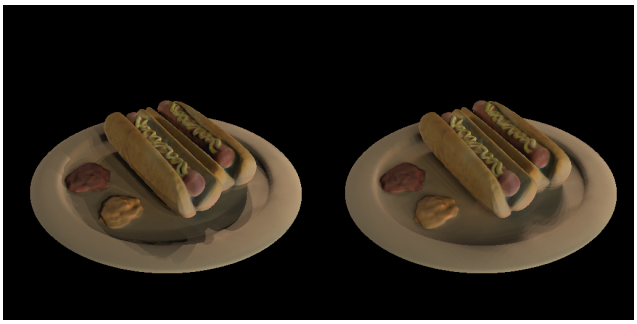
				Novel View Synthesis			**Relighting**		
Exp.	NEM	Smooth.	Upsamp.	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
I	✓			33.274	0.951	0.067	25.597	0.919	0.095
II		✓		32.844	0.951	0.065	26.213	0.924	0.093
III	✓	✓		32.725	0.950	0.068	26.022	0.923	0.093
Ours	✓	✓	✓	-	-	-	26.319	0.927	0.085

Table 2. **Ablation Studies.** The metrics are evaluate on hotdog scene over 8 uniformly sampled validation views. The relighting metric is averaged over 8 novel lighting conditions. Each metric's top three values are highlighted in red, orange, and yellow, respectively. NEM stands for neural environment map, Smooth. stands for albedo and BRDF smoothness, and Upsamp. stands for upsampling during inference.

Computation Time Comparison Our model calculates light visibility thousands of times faster than NeRFactor. NeRFactor requires more than 150 hours for computing the light visibility for 308 views, with 512 point light sources for each. In our model, it just takes five minutes. The reason is that NeRFactor must march points along a ray and integrate their sigma-volume, which is computationally expensive, whereas our model leverages the high speed of path tracing.

4.5. Ablation Studies

Smoothness By comparing Exp.I and Exp.III in Table 2 we can infer that, while limiting the performance of novel view synthesis, smoothness enhances the final relighting results. The albedo smoothness prevents our model from recovering an impure albedo with wrong information about shadows. The BRDF smoothness help to constrain rapid



(a) w/o Upsampling (b) w/ Upsampling

Figure 6. **Improving Shadow Estimation by Upsampling during Inference.** Upsampling helps efficiently generate a more realistic soft cast shadows without requiring additional training.

material changes.

Neural Environment Map Exp.II and Exp.III in Table 2 show the influence of using neural environment map (NEM) compared to using light probes with a fixed size. As can be seen, they achieve similar performance for the relighting. We believe that NEM has more potential for future work, for example, for predicting the environment map using Monte Carlo sampling.

Upsampling During Inference A simple and effective method for enhancing relighting performance without further training is to sample more light sources in the new surroundings. It is only enabled when a high-speed visibility calculation is performed. Table 2 shows that the upsampling can marginally enhance the relighting results. In our experiment, we sample 2048 light sources from a test light probes with a size of 512. We assume that all point light sources are surface light sources and sample four rays uniformly around each point light source. It involves recalculating the visibility, but as the visibility calculation takes less than 1.5 seconds per view for 2048 light sources, this is not an issue. The upsampling help our model to generate soft shadows as shown in Figure 6.

Normalizing Flows BRDF Model Physically based rendering requires a BRDF model that conserves energy:

$$\forall \omega_{\mathrm{i}}, \int_{\Omega} f_{\mathrm{r}}\left(\omega_{\mathrm{i}}, \omega_{\mathrm{o}}\right) \cos \theta_{\mathrm{o}} d \omega_{\mathrm{o}} \leq 1, \tag{9}$$

where ω_{i} is the incoming light direction, ω_{o} is the view direction and Ω is the unit hemisphere around the surface normal. However, representing BRDF directly with an MLP does not ensure this constraint. To incorporate this inductive bias into our BRDF model, we experimented with Normalizing Flows [11].

The main idea of Normalizing Flows is to model a complex distribution by applying a transformation to a simple distribution, e.g. a uniform or normal distribution. The change of variables formula from probability theory ensures that the transformed random variable has a well-defined probability density if the transformation is invertible and both the transformation and its inverse is differentiable [11]. Normalizing Flows methods differ by the way they model this transformation. One such example is the Real NVP method [5], where the transformation is simply defined as the following:

$$\begin{aligned} y_{1:d} &= x_{1:d} \\ y_{d+1:D} &= x_{d+1:D} \odot \exp\left(s\left(x_{1:d}\right)\right) + t\left(x_{1:d}\right), \end{aligned} \tag{10}$$

where x is a D dimensional input, y is a D dimensional output, s and t are functions from $R^d \rightarrow R^{D-d}$, $\odot$ is the element-wise product. "s" stands for scale and "t" stands for translation. These functions are represented by MLPs.

To apply Real NVP for energy conserving BRDF modelling, we need to condition it to the outgoing direction as we need the integral in Equation 9 to satisfy the constraint. To achieve that we simply input the conditioning variable to the scale and translation networks. We end up with the following transformation:

$$\begin{aligned} y_{1:d} &= x_{1:d} \\ y_{d+1:D} &= x_{d+1:D} \odot \exp\left(s\left(x_{1:d}, c\right)\right) + t\left(x_{1:d}, c\right), \end{aligned} \tag{11}$$

where c is the conditioning variable.

Another issue is that the incoming and outgoing directions are coupled together as they are represented by Rusinkiewicz coordinates by the MERL dataset [7], but we need to separate them since we condition our Normalizing Flow network on the outgoing direction. Therefore we first transform the Rusinkiewicz coordinates $(\phi_{\mathrm{d}}, \theta_{\mathrm{h}}, \theta_{\mathrm{d}})$ into spherical coordinates $(\phi_{\mathrm{i}}, \theta_{\mathrm{i}}, \phi_{\mathrm{o}}, \theta_{\mathrm{o}})$ using the implementation by Nielsen et al. [9]. However the materials in MERL are isotropic so we align the outgoing direction with the x-axis and eliminate ϕ_{o}. We end up with 3 degrees of freedom: $(\phi_{\mathrm{i}}, \theta_{\mathrm{i}}, \theta_{\mathrm{o}})$. We also need to condition our network on material latent codes, so we input them to scale and shift networks. Our final conditional transformation is:

$$\begin{aligned} y_{1:d} &= x_{1:d} \\ y_{d+1:D} &= x_{d+1:D} \odot \exp\left(s\left(x_{1:d}, \theta_{\mathrm{o}}, z_{\mathrm{BRDF}}\right)\right) \\ &\quad + t\left(x_{1:d}, \theta_{\mathrm{o}}, z_{\mathrm{BRDF}}\right). \end{aligned} \tag{12}$$

The base distribution is bivariate Gaussian distribution with an identity covariance matrix. We apply positional encoding [8] to θ_{o} before inputting it to the network. To achieve more complex distribution we stack 16 of this transformation, swapping dimensions of the input at each step [5]. Our normalizing flow network estimates

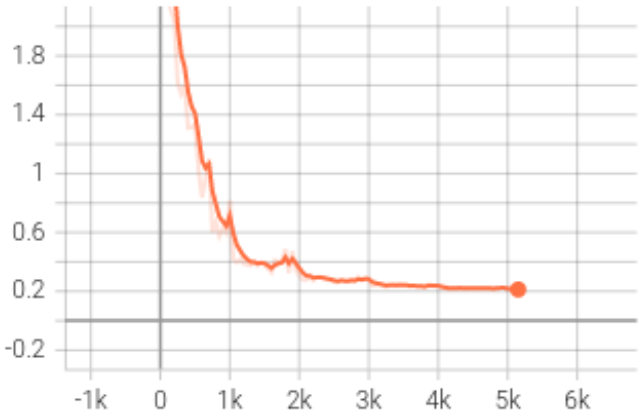


Figure 7. Normalizing Flows BRDF model training loss curve: We observe from this plot that our model cannot fit to the training data using Normalizing Flow BRDF model.

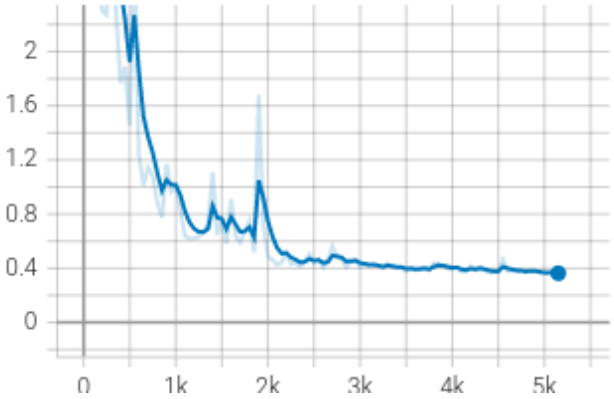


Figure 8. Normalizing Flows BRDF model validation loss curve: Our model fails to generalize for the MERL dataset using Normalizing Flow BRDF model.

$f_r(\omega_i, \omega_o)\cos\theta_o$ by applying inverse of the transformation to the incoming direction coordinates. Therefore the loss is L2-loss between the output of our network and the achromatic reflectance obtained from MERL(see Section 3.3) multiplied by $\cos\theta_o$.

We could not get a successful result from our experiments with our normalizing flow BRDF model as can be seen in Figure 7. The loss cannot be improved further below 0.2, which is too high considering we use L2-loss. As expected, our model does not generalize as seen in validation loss plot depicted in Figure 8.

We also integrated this BRDF model with our method to get qualitative results. In Figure 9, we observe that we fail to get different z_{BRDF} for different materials and as a result our albedo prediction is also not reasonable, since our specular BRDF cannot generalize.

We conclude from our normalizing flows experiments

Figure 9. Qualitative result from our method using the normalizing flows BRDF model(Left to right: ground-truth image, rendered image by our method, albedo, z_{BRDF}).

that our reasons for failure in modelling a normalizing flow BRDF may be numerical instabilities encountered during training and the fact that we cannot utilize Rusinkiewicz coordinates as they make learning specular highlights easier [12].

5. Future Work

In the future work we will further utilize differentiable Mitsuba2 path tracer and differentiable marching cubes algorithms to make our optimization end-to-end. Instead of sampling fix incoming light direction from the environment map, we will improve the sampling strategy with Monte Carlo sampling. We will continually explore a physic-based BRDF model. We expect that they would result in improved factorization and relighting.

6. Conclusion

We proposed a novel model that factorizes shape, material, and lighting of a scene, given multi-view images and poses under one unknown illumination. The main feature of our model is the integration of implicit and explicit 3D geometry representation. We leverage implicit SDFs to reconstruct a precise geometry and explicit meshes to conduct physic-based surface intersections and light visibility estimates with high efficiency. Thus, our model can recover plausible shape, reflectance, and environment lighting and predict convincing free-viewpoint view synthesis under arbitrary illuminations. We demonstrate that our model outperforms the previous works and achieves the state-of-the-art results.

References

[1] Lynton Ardizzone, Carsten Lüth, Jakob Kruse, Carsten Rother, and Ullrich Köthe. Guided image generation with conditional invertible neural networks, 2019. 2

[2] Jonathan T. Barron and Jitendra Malik. Shape, illumination, and reflectance from shading. 2020. 5

[3] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T. Barron, Ce Liu, and Hendrik P. A. Lensch. Nerd: Neural reflectance decomposition from image collections, 2020. 2

[4] Zhe Chen, Shohei Nobuhara, and Ko Nishino. Invertible neural brdf for object inverse rendering, 2020. 2

[5] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp, 2016. 2, 7

[6] William Lorensen and Harvey Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM SIGGRAPH Computer Graphics*, 21:163–, 08 1987. 1

[7] Wojciech Matusik, Hanspeter Pfister, Matt Brand, and Leonard McMillan. A data-driven reflectance model. *ACM Transactions on Graphics*, 22(3):759–769, July 2003. 1, 3, 7

[8] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 5, 7

[9] Jannik Boll Nielsen, Henrik Wann Jensen, and Ravi Ramamoorthi. On optimal, minimal brdf sampling for reflectance acquisition. *ACM Trans. Graph.*, 34(6), oct 2015. 7

[10] Merlin Nimier-David, Delio Vicini, Tizian Zeltner, and Wenzel Jakob. Mitsuba 2: A retargetable forward and inverse renderer. *ACM Trans. Graph.*, 38(6), nov 2019. 1, 2

[11] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. 2019. 2, 7

[12] Szymon Rusinkiewicz. A new change of variables for efficient BRDF representation. In *Rendering Techniques (Proc. Eurographics Workshop on Rendering)*, June 1998. 3, 8

[13] 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, 2020. 2

[14] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction, 2021. 1, 2

[15] Xiuming Zhang, Pratul P. Srinivasan, Boyang Deng, Paul Debevec, William T. Freeman, and Jonathan T. Barron. NeRFactor. *ACM Transactions on Graphics*, 40(6):1–18, dec 2021. 2, 3, 5, 6