VQ-NeRF: Neural Reflectance Decomposition and Editing with Vector Quantization

Hongliang Zhong, Jingbo Zhang, and Jing Liao*

***Abstract*—We propose VQ-NeRF, a two-branch neural network model that incorporates Vector Quantization (VQ) to decompose and edit reflectance fields in 3D scenes. Conventional neural reflectance fields use only continuous representations to model 3D scenes, despite the fact that objects are typically composed of discrete materials in reality. This lack of discretization can result in noisy material decomposition and complicated material editing. To address these limitations, our model consists of a continuous branch and a discrete branch. The continuous branch follows the conventional pipeline to predict decomposed materials, while the discrete branch uses the VQ mechanism to quantize continuous materials into individual ones. By discretizing the materials, our model can reduce noise in the decomposition process and generate a segmentation map of discrete materials. Specific materials can be easily selected for further editing by clicking on the corresponding area of the segmentation outcomes. Additionally, we propose a dropout-based VQ codeword ranking strategy to predict the number of materials in a scene, which reduces redundancy in the material segmentation process. To improve usability, we also develop an interactive interface to further assist material editing. We evaluate our model on both computer-generated and real-world scenes, demonstrating its superior performance. To the best of our knowledge, our model is the first to enable discrete material editing in 3D scenes.**

I. Introduction

Decomposing a scene into its constituent geometry, material, and lighting properties holds immense significance across various applications in the fields of computer vision and graphics, including scene relighting and appearance editing [1]–[5]. This challenging task, commonly referred to as inverse rendering [6]–[8], is inherently ill-posed due to the complex interplay between an object's observed color and its underlying lighting, material, and geometry attributes. For instance, the appearance of blackness in object renderings may be attributed to either insufficient lighting or dark material color. To overcome this inherent ambiguity, traditional methods for inverse rendering often incorporate additional constraints during the process of reflectance decomposition, such as controlled illumination [9], [10] or visual priors learned from 2D images [11]–[13]. However, such constraints significantly limit the applicability of the methods in real-world scenarios, where materials and illuminations are diverse and uncontrollable. In contrast, a more practical and universal strategy for inverse rendering is to introduce multi-view constraints and model the scene in a view-consistent representation [14], [15]. This approach allows the model to analyze the appearance of the scene from multiple viewpoints, which can help to disambiguate the influence of lighting, material, and geometry, resulting in improved accuracy of reflectance decomposition.

With the development of Neural Radiance Field (NeRF) [16], recent reflectance decomposition methods (e.g., Bi et al. [17], Zhang et al. [3], Boss et al. [4], and Srinivasan et al. [18]) begin to adopt it as a 3D representation and introduce multi-view constraints. Briefly, these neural reflectance decomposition methods learn a continuous field that maps the spatial coordinates of the scene to corresponding reflectance factors represented by a Spatially-Varying Bidirectional Reflectance Distribution Function (SV-BRDF) [19]. Thanks to the powerful modeling and rendering capabilities of the neural implicit field, these methods can capture the subtle characteristics in the texture and geometry from different views of the scene, resulting in more accurate decomposition results in the inverse rendering task. However, their continuous representation of BRDF attributes conflicts with reality, where objects are typically composed of discrete types of materials such as wood, plastic, metal, and others. The absence of discretization often leads to noisy decomposition within individual materials. As shown in the upper row of Fig. 1, the predicted specular attributes for the *bronze* balls vary significantly from location to location. Furthermore, the non-discrete modeling makes selecting specific materials for editing challenging, which in turn complicates appearance editing. Even with the help of the Meanshift clustering, as illustrated in Fig. 1, selecting the entire ball with the *bronze* material and editing it into a new *silver* material remains a challenge for such continuously decomposed materials.

To address the aforementioned issue, we propose VQ-NeRF, a novel neural reflectance decomposition framework based on Vector Quantization (VQ) [20], [21]. Our framework comprises a continuous branch and a discrete branch. The continuous branch predicts a 3D implicit field of reflectance factors under multi-view constraints, while the discrete branch employs the VQ mechanism to quantize the continuous reflectance field into a limited number of VQ codewords, resulting in a discrete segmentation map for different materials. These two branches are jointly optimized, with the VQ clustering in the discrete branch effectively constraining the reflectance prediction of the continuous branch to be more compact, approaching VQ codewords, and thus suppressing prediction noise, as illustrated in the lower row of Fig. 1. In turn, the adjusted prediction results in the continuous branch assist the discrete branch in learning more accurate codewords, enhancing material clustering performance.

Moreover, our VQ-NeRF framework enables the user to

*: corresponding author.

H. Zhong, J. Zhang and J. Liao are with Department of Computer Science, City University of Hong Kong. Email: hlzhong2-c@my.cityu.edu.hk, jbzhang6-c@my.cityu.edu.hk, jingliao@cityu.edu.hk.

conveniently select and edit specific materials by providing a material segmentation map through the discrete branch, as shown in Fig. 2. To prevent the presence of redundant materials in the segmentation map, we further introduce a dropout-based codeword ranking strategy to our VQ scheme. By sorting the material codewords according to their importance, our method can eliminate lower-ranked redundant codewords, ensuring that the number of predicted materials is appropriate for the scene complexity and facilitating user selection and editing of specific materials. Additionally, to support intuitive material editing in 3D scenes, we have built an interactive User Interface (UI). In this interface, users can view the segmentation map produced by the discrete branch from any angle and click on the corresponding areas to specify the materials to be edited. Our continuous branch then performs neural rendering with the edited materials and presents the results in different views. This allows users to have a more intuitive and interactive experience when editing materials in 3D scenes.

Thanks to the well-designed two-branch framework with VQ mechanism and dropout-based ranking strategy, our VQ-NeRF significantly enhances the accuracy of reflectance decomposition and empowers efficient material editing. We evaluated our method on both Computer-Generated (CG) scenes and real-world scenes, and demonstrated its superior performance in multiple tasks including scene reconstruction, reflectance decomposition, material editing, and scene relighting.

To sum up, our contributions are three folds:

- We propose VQ-NeRF, the first method that incorporates the VQ mechanism to discretize reflectance decomposition, thereby enhancing decomposition accuracy and facilitating material editing.
- We introduce a dropout-based VQ codeword ranking strategy that automatically determines the number of materials in an arbitrary scene, eliminating redundancy in VQ-predicted materials.
- We develop an interactive user interface that enables convenient material editing of 3D scenes in a view consistent manner.

II. Related Work

A. Traditional Reflectance Decomposition

Considering the inherent ambiguity during inverse rendering, classical methods usually require additional constraints to assist the reflectance decomposition process of the model [13], [22]–[24]. For example, LSR-BRDF [9] and CM-BRDF [10] simplify the reflectance computation by capturing scenes in a controllable lighting environment. Although they achieve plausible decomposed results in their specific experimental scenes, they cannot perform reasonable reflectance estimation for realistic scenarios with arbitrary and uncontrollable illumination. In contrast to constrain the lighting conditions, SA-CNN [11] adopts a deep convolution network to learn visual priors from planar material data and performs SV-BRDF estimation in the image space. Benefiting from the data-driven priors, this method achieves 2D reflectance decomposition under casual illumination. Nonetheless, it struggles to handle reflectance decomposition of 3D objects due to the lack of 3D priors. By contrast, CASCADE-CNN [12] trains a cascaded network on a more general 3D dataset containing multi-view material factors rendered from 3D objects using a complex SV-BRDF. Thanks to the enhanced model and 3D priors in the training data, this method enable to infer reflectance factors from rendered images of 3D objects. Still, it fails to produce view-consistent decomposed factors since the inference of each view is performed independently. To introduce multi-view constraints, MVGCR [14] reconstructs polygon meshes of the scene with Multi-View Stereo (MVS) [25], and uses it as a 3D representation to perform inverse rendering. Although such method realizes view-consistent reflectance decomposition, the low-fidelity mesh representation seriously limits its performance.

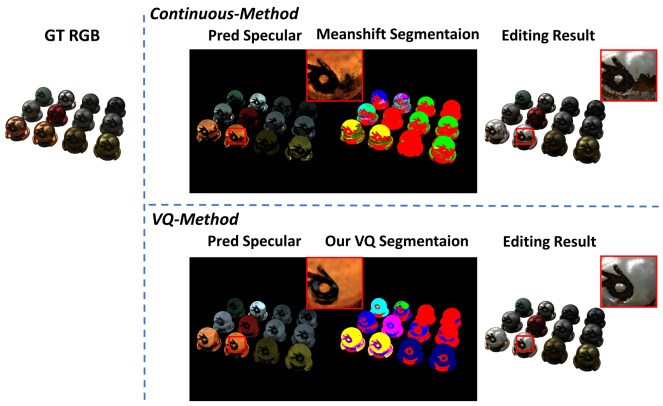

Fig. 1. Conventional neural reflectance decomposition methods (upper row) often predict noisy BRDF attributes for individual materials due to the absence of material discretization. This continuous representation also presents challenges for specific material editing. In contrast, our VQ-NeRF approach (lower row) incorporates the VQ mechanism to discretize reflectance decomposition, which suppresses prediction noise and facilitates material editing.

B. Neural Reflectance Decomposition

Inspired by the great success of the emerging NeRF and its variants [16], [26], [27] in 3D scene modeling, recent methods of reflectance decomposition [4], [18], [28], [29] attempt to leverage neural implicit fields [26], [30] as 3D representations to provide multi-view constraints during inverse rendering. For instance, NeRFactor [3] utilizes multiple implicit fields to model the scene geometry, albedo, and BRDF identity, respectively. Benefiting from the powerful representation capability of neural implicit fields in view-consistent modeling of materials, NeRFactor demonstrates promising results compared to traditional methods. However, since NeRFactor predicts specular reflections through a pre-trained network, its decomposition is largely constrained by the distribution bias of the pre-training data. Similarly, Neural-PIL [31] adopts a pre-trained network for material prediction and a pre-trained network for lighting estimation. This design allows Neural-PIL to predict different lighting in different views and thus achieve reflectance decomposition for scenes rendered in varying illuminations. Nonetheless, it suffers the

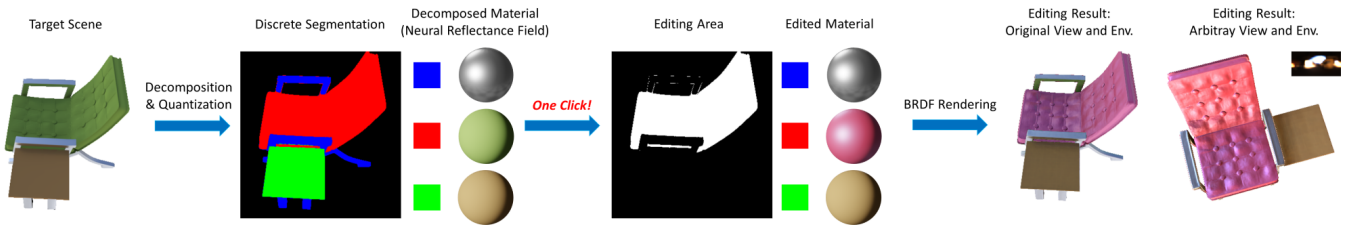

Fig. 2. We propose VQ-NeRF, which incorporates the VQ mechanism to discretize reflectance decomposition. This enables efficient and view-consistent material selection and editing.

same limitation as NeRFactor, as the bias of pre-training data seriously limits its decomposition performance. By contrast, NeILF [32] introduces an additional implicit lighting field to predict a corresponding lighting intensity for each surface point of the object. In this way, it is theoretically able to model the indirect lighting, leading to enhanced processing of complex illumination. However, due to the lack of sufficient restrictions, during the actual material decomposition process, this lighting field may be confused with the material fields, causing the lighting colors to be absorbed into the material fields. Unlike previous methods that employ volume rendering during optimization, NVDIFFREC [33] and NVDIFFRECMC [34] incorporate traditional rasterization-based rendering into their framework to accelerate computation by extracting scene meshes from their signed distance fields. However, explicit rasterization can easily lead to visible artifacts in rendered images, such as stretched geometry and blurry textures, which in turn affects the decomposition quality of materials, especially for some geometrically complex scenes. Besides, to best of our knowledge, all of existing methods for neural reflectance decomposition focus on continuous BRDF estimation, which is not conducive to material selection and editing, and conflicts with reality, because objects in real scenes are usually composed of discrete types of material. Therefore, we design a two-branch neural reflectance field based on the VQ mechanism to achieve discrete BRDF material decomposition.

III. Method

The pipeline of our VQ-NeRF approach is illustrated in Fig. 3. We first use a NeRF model to reconstruct the scene geometry and extract geometry components, such as surface normals and coordinates. Next, we jointly optimize a continuous branch network and a discrete branch network to perform reflectance decomposition and VQ-based material discretization, respectively. The continuous branch learns a neural reflectance field that maps spatial coordinates of the scene to corresponding reflectance factors represented by SV-BRDF, including diffuse, specular, and roughness, as well as an environment map. Meanwhile, the discrete branch employs the VQ mechanism to quantize the continuous reflectance field into a limited number of VQ codewords, yielding a material segmentation map. Additionally, we apply a dropout-based codeword ranking strategy to the discrete branch to reduce quantization redundancy. With the view-consistent segmentation map in an arbitrary rendering view, users can easily select a specific material for editing and produce the edited scene using BRDF rendering.

A. Geometry Reconstruction

The inputs to our method is a set of posed images I_i, which are captured from a 3D scene under natural illumination. To perform inverse rendering, we first use a NeRF network f_g to reconstruct the scene geometry. Similar as previous work, volume rendering [16] is employed to accumulate the color $C_v(\boldsymbol{r})$ in NeRF:

$$\begin{cases} C_v(\boldsymbol{r}) = \int_{t_n}^{t_f} T(t)\sigma(\boldsymbol{r}(t))c(\boldsymbol{r}(t), \boldsymbol{d})dt, \\ T(t) = \exp(-\int_{t_n}^{t} \sigma(\boldsymbol{r}(s))ds), \end{cases} \tag{1}$$

where $\boldsymbol{r}(t) = \boldsymbol{o} + t\boldsymbol{d}$ is the spatial points sampled on the camera ray emitted from the origin $\boldsymbol{o}$ in the direction $\boldsymbol{d}$. t_n and t_f represent the bounds of near and far sampling. $\sigma(\boldsymbol{r}(t))$ and $c(\boldsymbol{r}(t), \boldsymbol{d})$ indicate the predicted density and color of the sampled point $\boldsymbol{r}(t)$, respectively.

To train this network, we minimize the L_2 loss between the rendered color $C_v(\boldsymbol{r})$ and the pixel color $C_{gt}(\boldsymbol{r})$ in the input images. After training, we can extract the geometry components from the reconstructed model f_g, including the coordinates $\boldsymbol{p}$ of surface points and the associated surface normal $\boldsymbol{N}(\boldsymbol{p})$ [3], [4].

B. Continuous Branch

We construct a neural reflectance field in the continuous branch to map surface coordinates $\boldsymbol{p}$ into BRDF attributes. The reflectance field involves an encoder f_e and a decoder f_d^c, both of which are composed of MLP networks. The encoder maps the input spatial coordinates $\boldsymbol{p}$ into latent material vectors $\boldsymbol{z}(\boldsymbol{p})$ and the decoder predicts the BRDF material factors according to the latent vectors $\boldsymbol{z}(\boldsymbol{p})$:

$$\begin{cases} \boldsymbol{k}_d(\boldsymbol{p}), \boldsymbol{k}_m(\boldsymbol{p}), \boldsymbol{k}_r(\boldsymbol{p}) = f_d^c(\boldsymbol{z}(\boldsymbol{p})), \\ \boldsymbol{z}(\boldsymbol{p}) = f_e(\boldsymbol{p}), \end{cases} \tag{2}$$

where $\boldsymbol{k}_d$, $\boldsymbol{k}_m$, and $\boldsymbol{k}_r$ indicate the basecolor, metallic, and roughness, respectively [32], [33]. Subsequently, the basecolor $\boldsymbol{k}_d$ and metallic $\boldsymbol{k}_m$ are further converted into the diffuse attribute $\boldsymbol{k}_\alpha$ and the specular attribute $\boldsymbol{k}_s$ for the following rendering [31]:

$$\begin{cases} \boldsymbol{k}_\alpha(\boldsymbol{p}) = \boldsymbol{k}_d(\boldsymbol{p}) \cdot (1 - \boldsymbol{k}_m(\boldsymbol{p})), \\ \boldsymbol{k}_s(\boldsymbol{p}) = \boldsymbol{k}_d(\boldsymbol{p}) \cdot \boldsymbol{k}_m(\boldsymbol{p}). \end{cases} \tag{3}$$

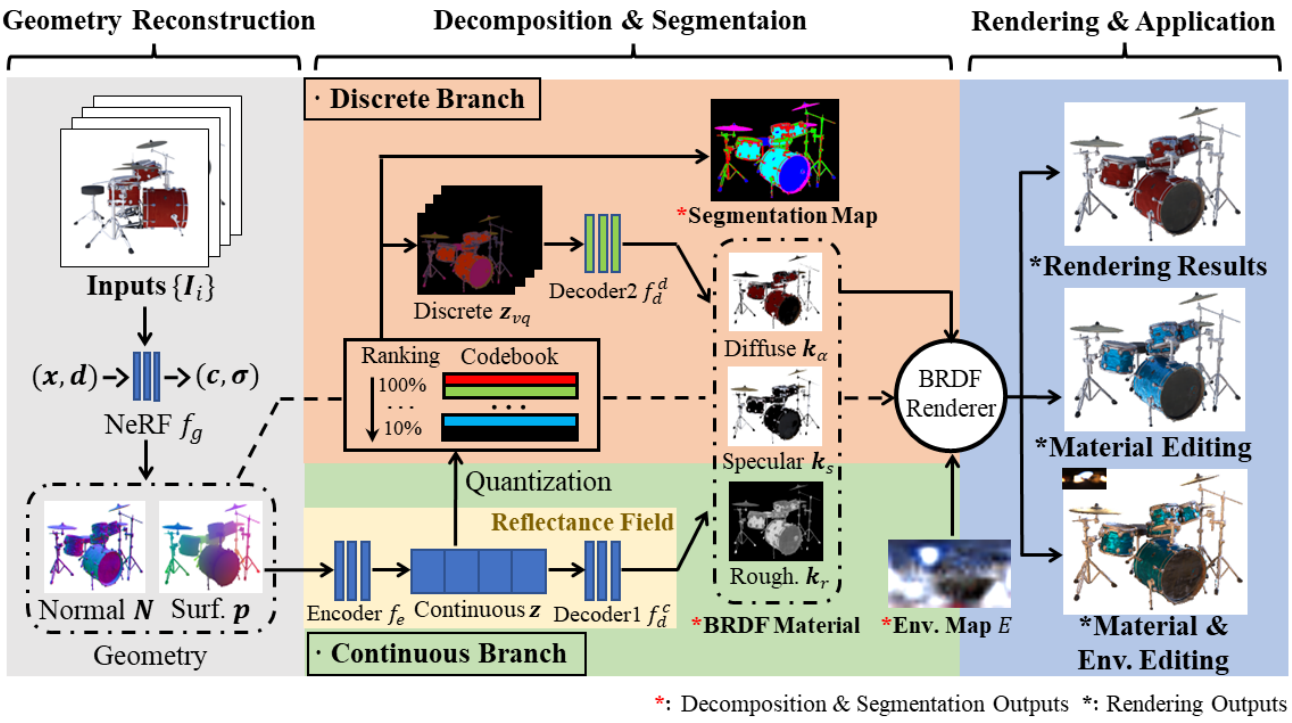

Fig. 3. The pipeline of our VQ-NeRF, the outputs are marked by asterisks (*). We first take multi-view posed images as inputs and use a NeRF model (gray part) to reconstruct the scene geometry. Next, we apply a two-branch network for reflectance decomposition and material discretization. The continuous branch (green part) predicts the decomposed BRDF attributes, including diffuse, specular, and roughness, while the discrete branch (red part) uses the VQ mechanism to discretize reflectance factors. After optimization, a material segmentation map is generated, which enables us to easily select specific materials for editing.

To deduce the rendered color $C_r(\boldsymbol{p})$ of the surface point $\boldsymbol{p}$ from these predicted BRDF attributes, we adopt Microfacet BRDF model [35] as our BRDF renderer:

$$C_r(\boldsymbol{p}) = \int_{\Omega} L_i \cdot f_R(\boldsymbol{k}_\alpha, \boldsymbol{k}_s, \boldsymbol{k}_r; \boldsymbol{p}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o)(\boldsymbol{\omega}_i \boldsymbol{N}(\boldsymbol{p}))d\boldsymbol{\omega}_i, \tag{4}$$

where Ω represents the illumination sphere surrounding the scene. L_i denotes the incoming illumination from the i-th lighting source, which is sampled from a learnable environment map E. $f_R(\cdot)$ denotes the BRDF function. $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_o$ represent the lighting direction and viewing direction, respectively. $(\boldsymbol{\omega}_i \boldsymbol{N}(\boldsymbol{p}))$ indicates the angle between the surface normal $\boldsymbol{N}(\boldsymbol{p})$ and the lighting direction $\boldsymbol{\omega}_i$.

Due to the BRDF attributes $\boldsymbol{k}_d$, $\boldsymbol{k}_m$, and $\boldsymbol{k}_r$ are predicted from the continuous neural reflectance field, it conflicts with reality where the material distribution in the scene is discretized and regionalized. To solve this issue, we introduce a discrete branch in the following section to quantize the hidden vectors predicted by the continuous branch and produce discretized material attributes.

C. Discrete Branch

1) Vector Quantization: In this section, we construct a discrete branch combined the VQ mechanism [20], [21] for material discretization, which facilitates material selection and editing. For each latent material vector $\boldsymbol{z}$ produced by the continuous branch, VQ mechanism matches it with a most similar codeword $\boldsymbol{z}_{vq}$ selected from its trainable codebook:

$$\begin{cases} u = \underset{i}{\operatorname{argmin}} |\boldsymbol{e}_i - \boldsymbol{z}|^2, \ i \in 1, ..., M, \\ \boldsymbol{z}_{vq} = \operatorname{sg}(\boldsymbol{e}_u - \boldsymbol{z}) + \boldsymbol{z}, \end{cases} \tag{5}$$

where M denotes the length of the VQ codebook. $\boldsymbol{e}_i$ repesents the i-th codeword. $\operatorname{sg}(\cdot)$ indicates the stop gradient operation. During the VQ clustering, both $\boldsymbol{z}$ and $\boldsymbol{z}_{vq}$ are normalized to the unit sphere for computational convenience.

Consequently, we employ another decoder f_d^d to infer discrete material attributes from the quantized latent vectors $\boldsymbol{z}_{vq}$. With these discrete material attributes, a similar BRDF rendering process can be performed to infer the rendered color $C_{r,d}(\boldsymbol{p})$ in arbitrary views by using Eq. 4. Besides, we can easily deduce a material segmentation map according to these discrete materials for facilitating material selection and editing. Notably, since both $\boldsymbol{z}$ and $\boldsymbol{z}_{vq}$ are predicted solely from the surface points $\boldsymbol{p}$, the material segmentation map is view-consistent and can be deduced in any desired view.

2) Dropout-based Ranking: Although VQ achieves the discretization of continuous materials, how to determine the length of the VQ codebook M is still a problem due to the uncertainty of the number of materials in a scene. To determine the codebook length automatically and eliminate material redundancy, we introduce a dropout-based codeword ranking strategy during the reflectance decomposition. Specifically, we first set an initial length M_0 for the codebook and assign a dropout rate to each codeword. The dropout rates are set in ascending order, increasing linearly from 0

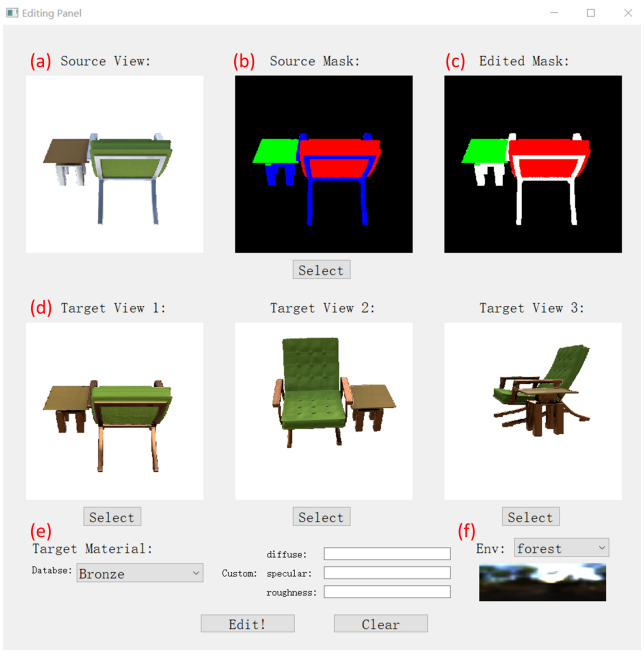

Fig. 4. Our UI for interactive material selection and editing. The reconstructed image and segmentation map of an arbitrary view are presented in (a), (b) and (c). Users can click in (b) to specify the editing area, and assign the target material in (e). The lighting of the scene can also be adjusted in (f). After configuring all the settings, the user can start the model by clicking the 'Edit!' button. The edited results are shown in (d) for visualization.

to around 0.7. During optimization, the codewords will be randomly dropped out from the codebook according to their assigned dropout rates. The frontier codewords, which possess lower dropout rates, are more likely to participate in model optimization and loss computation. As a result, to minimize the overall reconstruction loss of the discrete branch, the frontier codewords are considered crucial materials that significantly impact the reduction of the reconstruction error. Through this importance-driven prediction, the codewords are automatically sorted by importance after training.

Then, we perform multiple evaluations to determine the appropriate length M for the codebook. Specifically, we conduct the decomposition and BRDF rendering process in the discrete branch multiple times, using the first k codewords from the codebook, and compute the reconstruction error err_k at each evaluation. As the codewords have been sorted by importance, the curve formed by these reconstruction errors exhibits a trend of rapid decline followed by flattening. Therefore, we can determine the length M by finding the point on the curve where the flattening occurs, i.e., the first point satisfying the condition $|err_k - err_i| <= \epsilon, i \in k+1, ..., M_0$, from $k = 1$ to M_0. Here, ϵ is a constant for flattening determination, which is empirically set as 0.002 in practice.

D. Two-branch Joint Training

To encourage mutual benefit between the continuous and discrete branches, we use a joint training strategy. During training, we adopt compound objectives to constrain our two-branch reflectance decomposition framework, including a two-branch reconstruction loss, a VQ loss, a smooth loss, a similarity loss, and a Lambertian loss.

Specifically, we separately calculate the L_2 losses between the rendered color C_r ($C_{r,d}$) and the pixel color C_{gt} in the continuous and discrete branches as the reconstruction loss. Besides, to eliminate the influence of illumination on the VQ discretization process, we also calculate the reconstruction loss of the discrete branch in the chromaticity space: $L_{chr} = |\text{chr}(C_{r,d}) - \text{chr}(C_{gt})|^2$, where chr$(\cdot)$ is the transformation function from RGB space to chromaticity space. Therefore, the two-branch reconstruction loss is defined as $L_{rec} = w_1 L_{rec,c} + w_2 L_{rec,d} + w_3 L_{chr}$. Here, $L_{rec,c}$ and $L_{rec,d}$ are the L_2 losses in the continuous and discrete branches, respectively. w_1, w_2, and w_3 are constant parameters balancing between terms. The VQ loss is composed of two terms, defined as:

$$L_{vq} = |\boldsymbol{z}_{vq} - \text{sg}(\boldsymbol{z})|^2 + \lambda \cdot |\boldsymbol{z} - \text{sg}(\boldsymbol{z}_{vq})|^2, \tag{6}$$

where λ is a constant parameter. Additionally, we design a Lambertian loss such that surface points with high roughness are predicted to have low specular, which is consistent with real material relations. The Lambertian loss is formed as $L_{lam} = w_r \cdot \boldsymbol{k}_s$, where the weight $w_r = 2 \cdot \text{sg}(\boldsymbol{k}_r) - 1$ if $\boldsymbol{k}_r > 0.5$ else 0.

In practice, we find that a large area of the same material may be mistakenly divided into multiple pieces of similar materials during VQ clustering. To solve this problem, we introduce a smooth loss to achieve a color-aware constraint:

$$L_{sm} = \exp(-\alpha \cdot e_{chr}) \cdot (1 - \boldsymbol{z}_{vq,i}, \cdot \boldsymbol{z}_{vq,j}), \tag{7}$$

where the exponential component $e_{chr} = ||\text{chr}(C_{gt,i}) - \text{chr}(C_{gt,j})||_2^2$ if $||\text{chr}(C_{gt,i}) - \text{chr}(C_{gt,j})||_2^2 > \beta$, else $e_{chr} = 0$. α and β are constant parameters for value scaling and threshold clipping. $C_{gt,i}$ and $C_{gt,j}$ indicate the colors of adjacent surface points $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$. $\boldsymbol{z}_{vq,i}$ and $\boldsymbol{z}_{vq,j}$ are the corresponding clustering codewords.

Thus, the overall objective can be be expressed as $L_{all} = L_{rec} + w_4 L_{vq} + w_5 L_{lam} + w_6 L_{sm}$, where w_4, w_5, and w_6 are constant parameters.

E. User Interface

Furthermore, we develop a UI for interactive material selection and editing. As shown in Fig. 4, users enable render the reconstructed model in an arbitrary view (a) and obtain the corresponding segmentation map (b) inferred by our discrete branch. Then, they can click the segmentation map to select the areas to be edited, as shown in (c). By setting the target material and environment map on (e) and (f) from the associated databases, the edited 3D model with desired material and illumination will be re-rendered in the region (d) of the UI.

IV. Experiments

A. Setup

Experiment Data. To evaluate the performance of our method, we conduct reflectance decomposition experiments on both CG dataset and real dataset. Here, we collect five

scenes (*drums*, *hot-dog*, *ficus*, *lego*, and *metal-balls*) released by NeRFactor [3] and NeRF [16] as the CG dataset. Notably, due to these scenes lack the ground truth of specular and roughness, we construct three additional scenarios (*kitchen*, *chair*, and *blender*) for evaluation on specular and roughness components. The real dataset includes seven scenes captured by us (*rabbit*, *kettle*, *tools*, *shoes*, *wooden-chair*, *redcar*, and *lord-rabbit*) and three scenes collected from the DTU [36] dataset (*golden-sculpture*, *house*, and *dolls*).

Baseline Methods. We compared our VQ-NeRF to five state-of-the-art reflectance decomposition methods, including NeRFactor [3], Neural-PIL [31], NeILF [32], NVDIFFREC [33], and NVDIFFRECMC [34]. All of the experiments are conducted on author-released codes for compared methods.

Metrics. Following other works [3], [16], [33], we use Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS) as quantitative evaluation metrics in appearance reconstruction, reflectance decomposition, and scene relighting. Higher PSNR/SSIM scores and lower LPIPS scores indicate better quality. As for the evaluation of segmentation results, we follow [37] to measure segmentation accuracy using F1-score, Precision Rate (P) and Recall Rate (R) calculated under both micro and macro average. Higher scores for all of these metrics indicate better quality.

Implementation Details. In practice, we set the super-parameters in training loss as $w_1 = 0.2$, $w_2 = w_3 = w_4 = 1$, $w_5 = 0.001$, and $w_6 = 0.05$. The λ in Eq. 6 is set to 0.1. And for Eq. 7, we set $\alpha = 60$ and $\beta = 0.1$. M_0 is set to 8 for most of the scenes. But for the *tools* scene, the *kitchen* scene, and the scenes in the CG dataset, we set $M_0 = 15$, as their material compositions are more complicated. Our pipeline supports the use of various NeRF variants for geometry reconstruction. Specifically, we use NeuS [36] in our implementation. To achieve robust convergence, we employ VQ-EMA [38] instead of plain VQ in our discrete branch. Inspired by [39], [40], we additionally bake a residual into the reconstructed images. The residual baking is independent of reflectance decomposition, material editing, and scene relighting, but can bring richer details (such as intra-scene reflections) in appearance reconstruction. Considering the differences between the BRDF models adopted by different methods, the light-albedo scales of different methods are inconsistent. To solve this issue and perform a fair comparison, similar as previous methods [3], [33], [34], we normalize the decomposed materials and relighted images to match the average luminance of the reference via an indeterminable scale factor for each method.

B. Reconstruction and Reflectance Decomposition

We first evaluate the performance of our VQ-NeRF on both scene reconstruction and reflectance decomposition tasks. Table I and Fig. 5 display the quantitative and qualitative results produced by baseline methods and ours on the CG data realised by previous methods. Due to these data lack specular and roughness references, we only perform comparison on the basecolor and final reconstruction results in Table I and Fig. 5. As a supplement, we show comprehensive evaluation results on CG data, including specular and roughness components, in Table II and Fig. 6. Furthermore, we perform reconstruction and decomposition on the real data, and show the comparison in Table III and Fig. 7. As there is no ground truth for materials in real data, the quantitative comparison only contains the reconstruction scores. Obviously, compare to baseline methods, our method demonstrates superior performance in terms of scene reconstruction and reflectance decomposition tasks, both on the CG dataset and the real dataset.

TABLE I
RECONSTRUCTION AND REFLECTANCE DECOMPOSITION RESULTS ON THE CG DATASET PROVIDED BY PREVIOUS METHODS.

	Reconstruction			Basecolor		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
NVDIFFREC	33.193	0.968	0.021	25.006	0.926	0.080
NVDIFFRECMC	32.090	0.960	0.032	27.582	**0.953**	0.054
NeRFactor	31.736	0.947	0.037	26.452	0.942	0.057
Neural-PIL	26.366	0.899	0.084	24.298	0.890	0.112
NeILF	32.365	0.958	0.024	23.883	0.911	0.082
Ours	**35.390**	**0.975**	**0.014**	**29.475**	0.949	**0.046**

Specifically, the baseline methods NVDIFFREC and NVDIFFRECMC utilize rasterization-based rendering on explicit mesh, leading to noticeable artifacts such as stretched geometry and blurry textures in their outputs. For example, in the *drums* scene depicted in Fig. 5, distinct noise is evident in their decomposed basecolor. Similarly, in the real-world *dolls* scene shown in Fig. 7, these methods miss out on capturing intricate features like the eyes, noses, and decorations on the *dolls* in their decomposed materials. In contrast, although the implicit baselines do not suffer from these shortcomings, they come with their own limitations. NeRFactor and Neural-PIL, both relying on pre-trained networks for BRDF prediction, are heavily influenced by the distribution bias of the pre-training data. As a result, their decomposition results struggle with correctly predicting colors for diverse materials. This is evident in the *metal-balls* scene illustrated in Fig. 5, where both methods fail to provide accurate color predictions. NeILF, on the other hand, lacks sufficient restrictions on illumination, leading to the absorption of lighting colors into the decomposed materials. This is evident in the *rabbit* scene shown in Fig. 7. Additionally, since all these baselines primarily focus on continuous BRDF estimation, their decomposed materials tend to exhibit considerable noise, as observed in the *kitchen* scene in Fig. 6.

In contrast to these methods, our VQ-NeRF gets rid of all those limitations. By leveraging a mutually beneficial two-branch pipeline for reflectance decomposition, the material components produced by our method are free from artifacts and present the most accurate colors. Besides, with VQ discretization and dropout-based ranking strategy, our method enables compact predicted materials and suppresses the noise in the decomposed factors. Therefore, our method achieves reasonable scene reconstruction and reflectance decomposition, which in turn facilitates subsequent material editing and illumination editing.

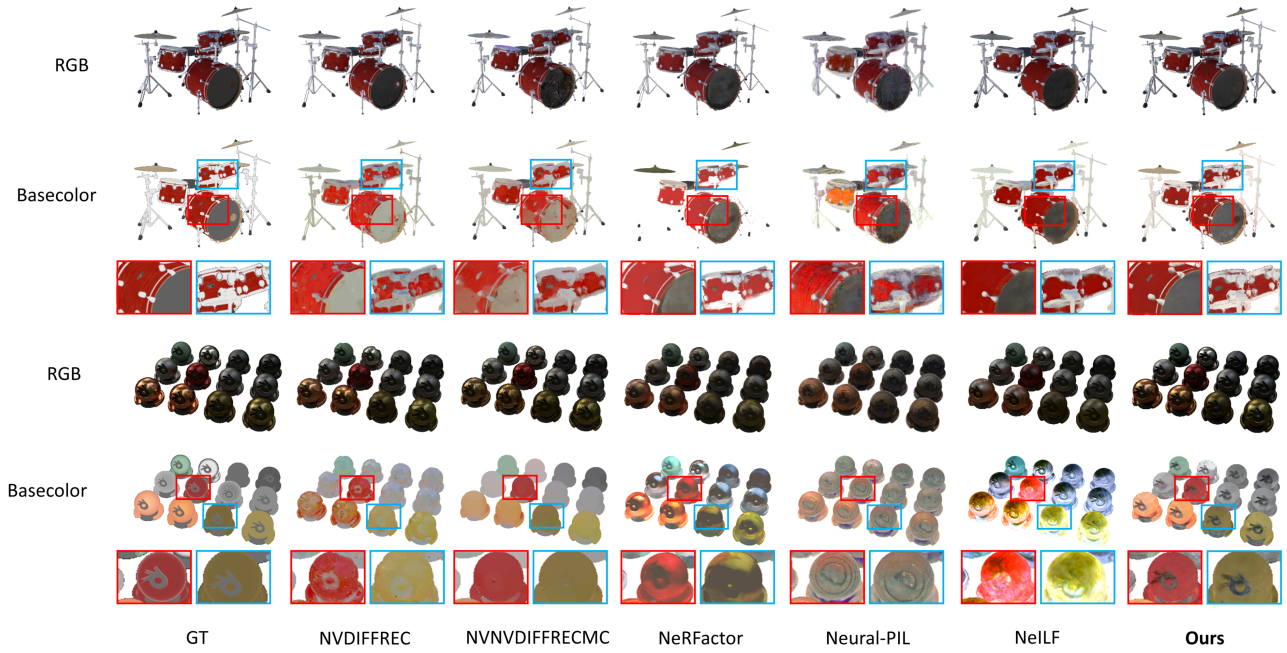

Fig. 5. Reconstruction and reflectance decomposition results on the CG dataset provided by previous methods. Obviously, the reflectance factors predicted by our model exhibit the most proper color and the least noise.

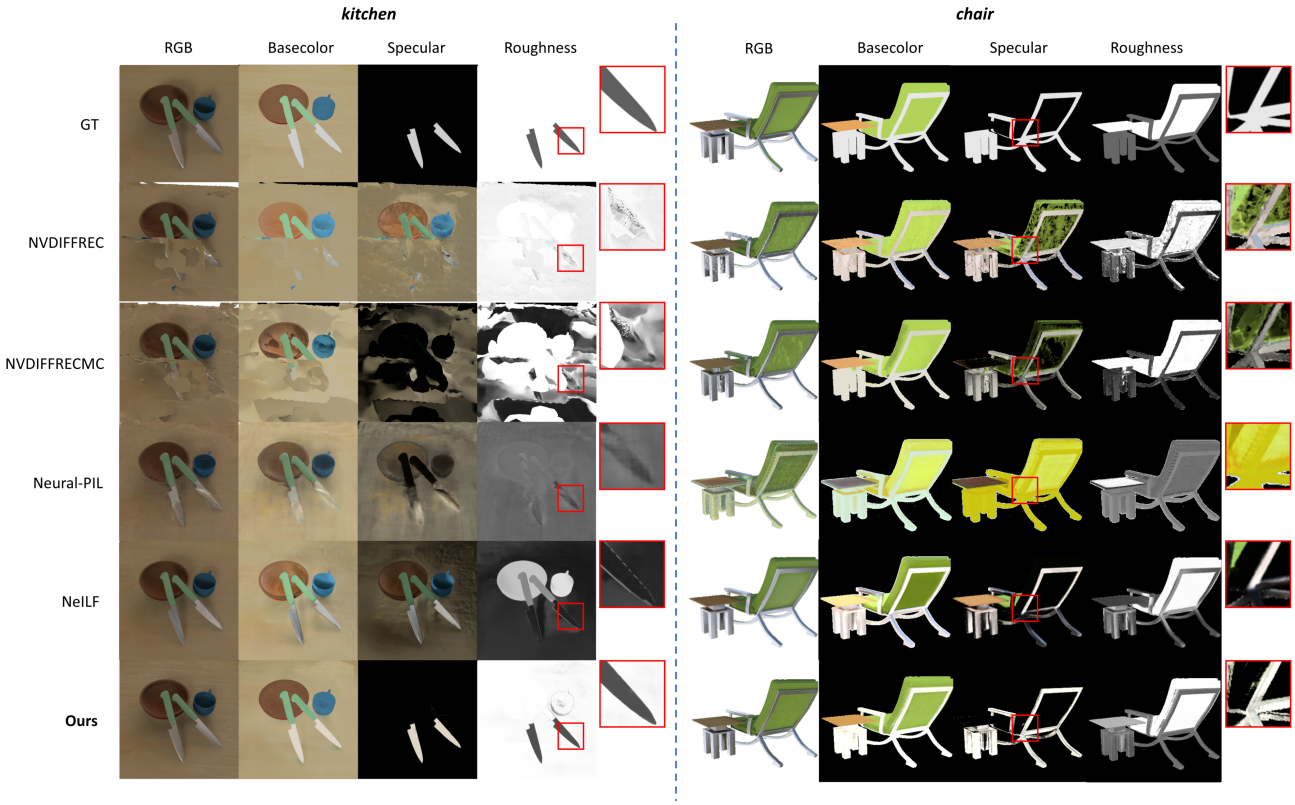

Fig. 6. Reconstruction and reflectance decomposition results on our CG dataset. The specular and roughness of NeRFactor are parameterized as latent codes in a network [3], [41] and cannot be extracted explicitly. So we exclude NeRFactor from this comparison. Evidently, our model achieves superior performance in the prediction of all three BRDF attributes.

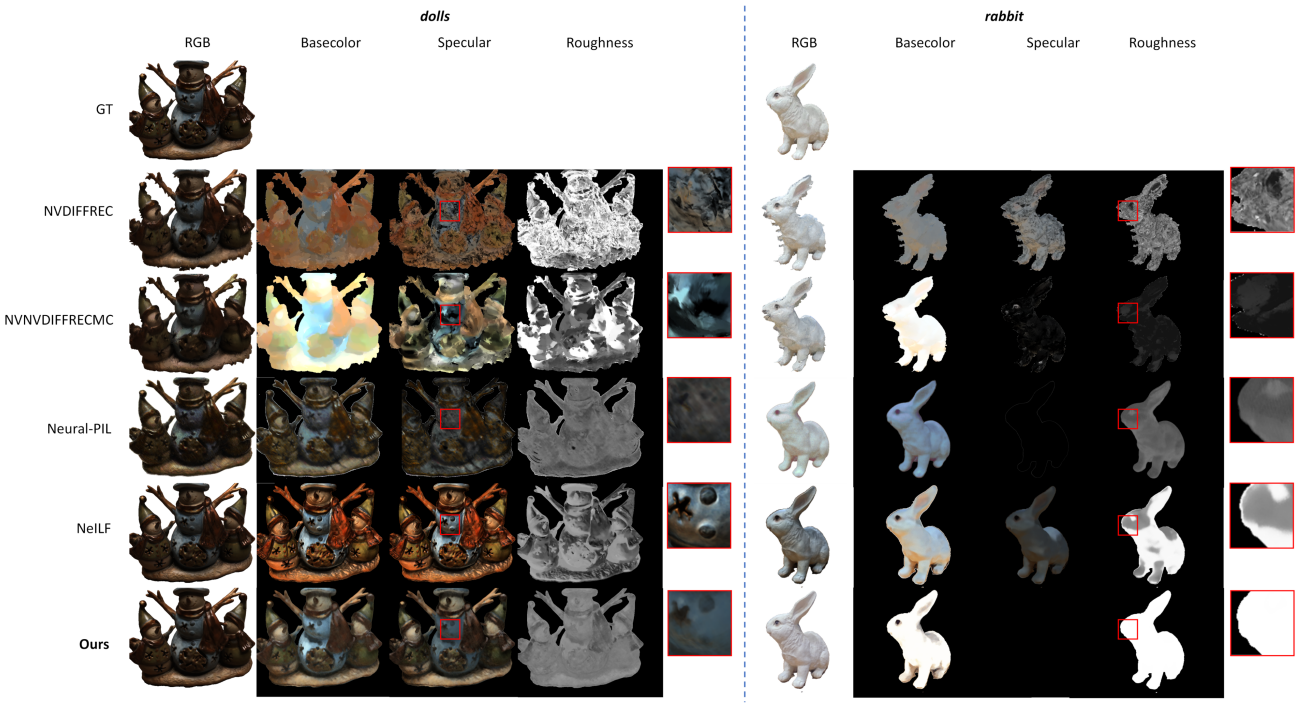

Fig. 7. Reconstruction and reflectance decomposition results on the real dataset. Our model achieves the best overall performance on the correctness and conciseness of material color, and our decomposition between individual BRDF attributes are also more accurate.

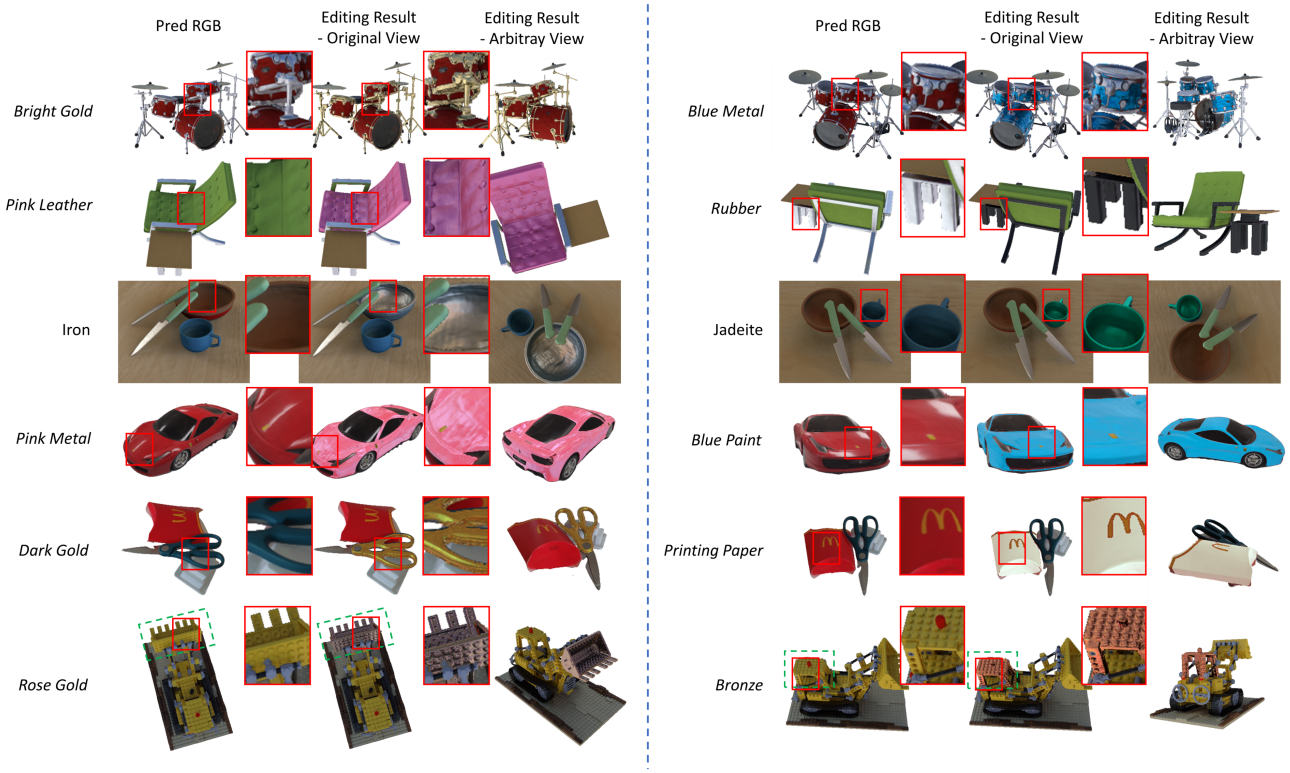

Fig. 8. Material editing results. With the help of our segmentation map, areas with the same material can be easily selected with just one or two mouse clicks and edited into a new material with specified BRDF parameters. Our model also supports local editing within bounding boxes, which are marked with green boxes in the *lego* case.

TABLE II
RECONSTRUCTION AND REFLECTANCE DECOMPOSITION RESULTS ON OUR CG DATASET

	Reconstruction			Basecolor			Specular			Roughness		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
NVDIFFREC	30.674	0.954	0.105	26.198	0.945	0.110	12.317	0.627	0.370	20.498	0.898	0.238
NVDIFFRECMC	30.135	0.937	0.134	27.352	0.945	0.130	17.939	0.736	0.262	18.361	0.849	0.241
Neural-PIL	27.966	0.903	0.147	22.472	0.914	0.153	11.577	0.620	0.346	12.229	0.814	0.365
NeILF	30.172	0.968	0.057	24.027	0.949	0.105	13.918	0.668	0.272	12.232	0.795	0.313
Ours	**35.470**	**0.978**	**0.041**	**28.860**	**0.962**	**0.072**	**26.258**	**0.966**	**0.059**	**27.882**	**0.972**	**0.069**

TABLE III
RECONSTRUCTION RESULTS ON THE REAL DATASET.

	Reconstruction		
	PSNR↑	SSIM↑	LPIPS↓
NVDIFFREC	27.232	0.887	0.106
NVDIFFRECMC	25.238	0.872	0.132
NeRFactor	24.523	0.881	0.133
Neural-PIL	25.848	0.865	0.133
NeILF	27.005	0.923	0.070
Ours	**32.745**	**0.944**	**0.066**

TABLE IV
RELIGHTING RESULTS ON THE CG DATASET. WE USE EIGHT LIGHTING PROBES PROVIDED BY NVDIFFRECMC FOR QUANTITATIVE EVALUATION.

	Relighting		
	PSNR↑	SSIM↑	LPIPS↓
NVDIFFREC	23.385	0.892	0.082
NVDIFFRECMC	26.961	0.919	0.064
NeRFactor	24.850	0.915	0.067
Ours	**27.515**	**0.931**	**0.044**

C. Material Editing

Since our VQ-NeRF is the first method to introduce discrete reflectance decomposition, it facilitates material selection and editing. The incorporation of the VQ mechanism not only suppresses decomposition noise but also supports the deduction of material segmentation maps in arbitrary views. Thanks to our dropout-based ranking strategy, the produced segmentation maps are compact and accurate. To achieve material editing, we first specify the desired material to be edited by selecting the corresponding area on the segmentation map. Subsequently, the selected material is replaced with new material with specified BRDF parameters, and the BRDF is applied to render the edited scene in arbitrary views. As shown in Fig. 8, all edits are made precisely in the corresponding areas, and obvious reflection changes such as highlight alternation are clearly visible in the edited model. In contrast to directly selecting the material to be edited in the segmentation map, other selection methods, such as bounding boxes, can also be combined with our segmentation map to support diverse local selection and editing, as shown in the *lego* case. Please refer to our video for more details.

D. Illumination Editing

We further performed illumination editing on both CG data and real data, and compared our method with baseline methods that support relighting. Table IV and Fig. 9 show the quantitative and visual relighting results on the CG data, respectively, while Fig. 10 shows the results on the real data. Thanks to our high-quality reflectance decomposition, our method yields the most realistic relighted images with the highest metric scores compared to baseline methods.

Specifically, NVDIFFREC and NVDIFFRECMC struggled to produce clear and reasonable relighting scenes due to their explicit geometric representation and unreliable material modeling. Models produced by such methods contain a large number of rough surfaces caused by the extruded geometry, as shown in Fig. 10, which seriously degrade the quality of scene relighting. Unlike NVDIFFREC and NVDIFFRECMC, NeRFactor can produce plausible geometry thanks to its implicit scene representation. However, its reflectance decomposition module is based on a pre-trained BRDF prediction network, which limits its ability to accurately model materials and illumination. This is why NeRFactor fails to produce reasonable relighting colors in many cases, such as the *metal-balls* scene in Fig. 9.

In contrast, our VQ-NeRF is able to generate reasonable relighting results, exhibiting proper color and obvious highlight reflections under different environmental lighting, as shown in the *drums* and *lego* scenarios of Fig. 9. Moreover, since our VQ-NeRF enables efficient material editing and scene relighting, our method also supports simultaneous material and lighting editing. As shown in Fig. 11, the left column shows the reconstructed result with the original lighting, while the right column shows the edited result with novel material and lighting.

E. Ablation Study

1) Two-Branch vs. Single-Branch: In our two-branch framework, the continuous branch outputs decomposition, while the discrete branch yields the segmentation map for material clustering. To validate the effectiveness of our two-branch design, we compare it with a single-branch option. Specifically, we use the discrete branch to simultaneously learn reflectance decomposition and VQ clustering, and as a result, it outputs both a segmentation map and BRDF factors after VQ discretization. As shown in Fig. 12, both single-branch and two-branch options can generate the correct segmentation map. However, the single-branch model produces completely flattened basecolor and cannot reconstruct subtle variations within a single material, such as the wooden texture, because the reflectance decomposition is strictly constrained by the VQ clustering. In contrast, in our two-branch framework, the VQ clustering constraint imposed by the discrete branch is a soft constraint, which facilitates the continuous branch in performing more discrete reflectance decomposition while still allowing for small variations within the same type of material

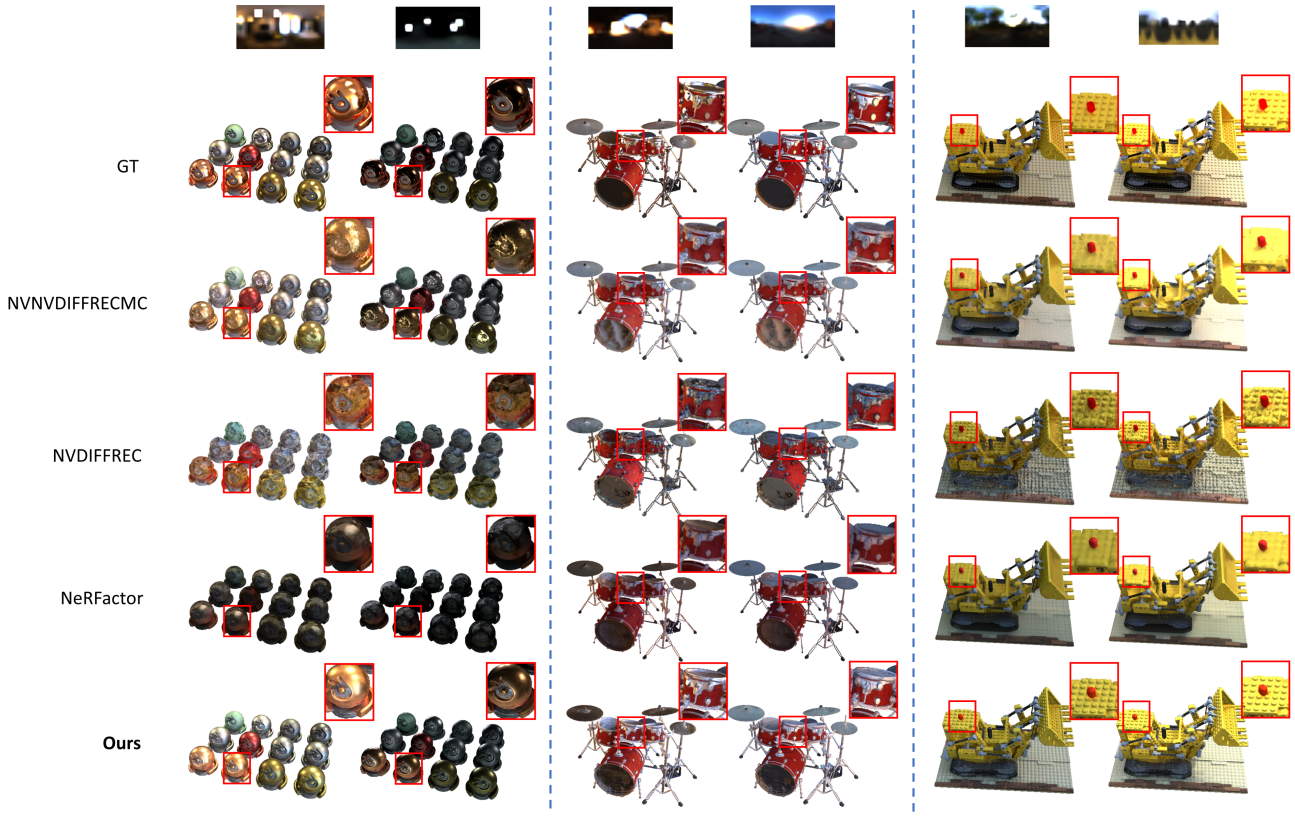

Fig. 9. Relighting results on the CG dataset. Evidently, our model produces realistic images with prominent highlights and clean appearance.

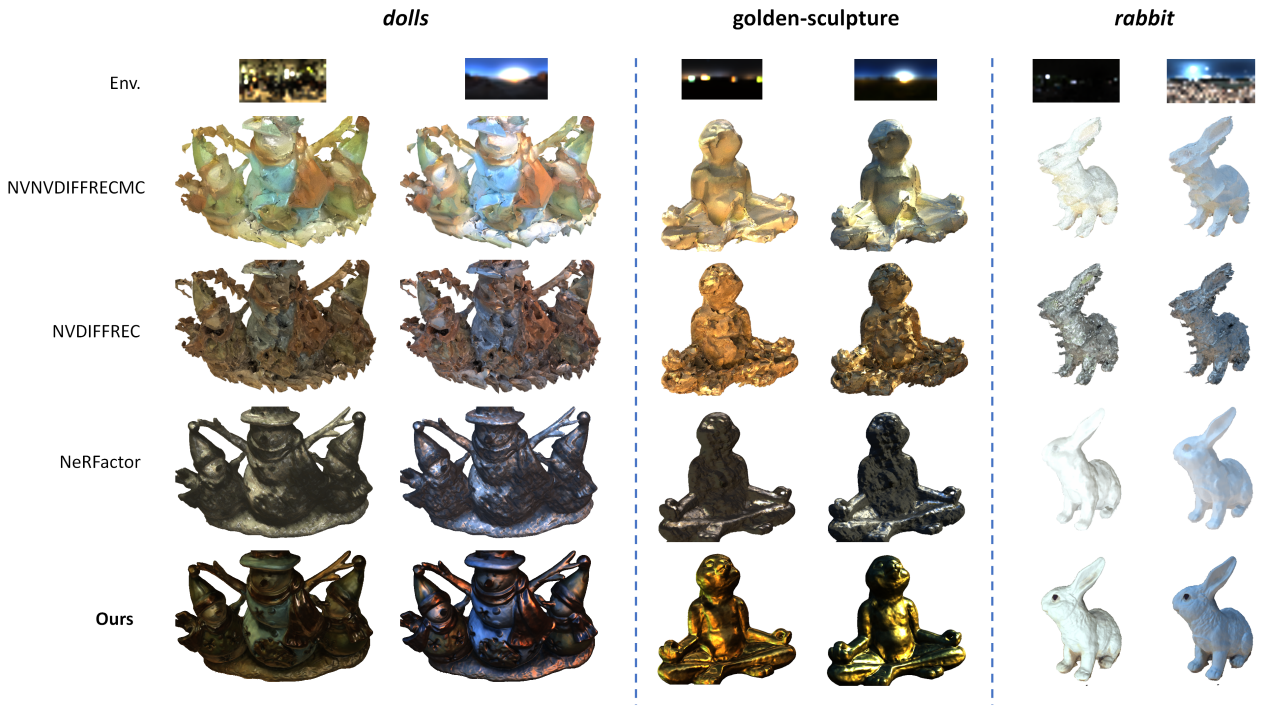

Fig. 10. Relighting results on the real dataset. In real-world scenes, our model can still produce realistic images with accurate colors and reflections.

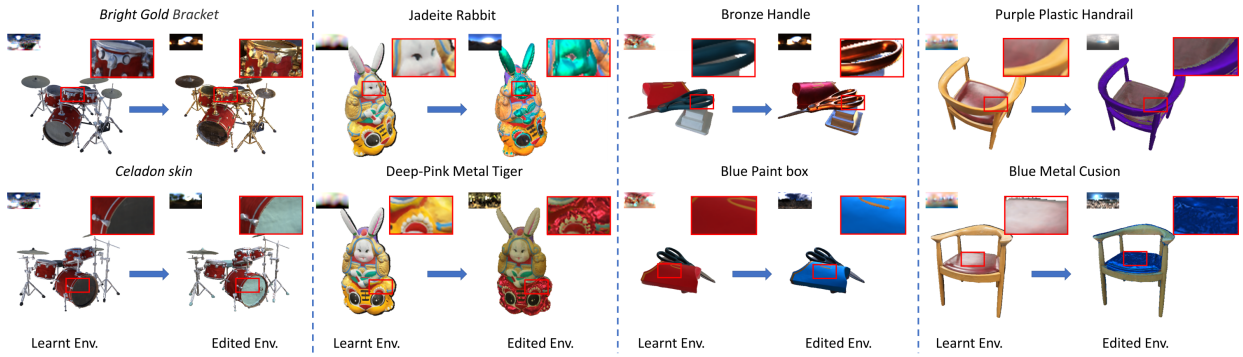

Fig. 11. Material-illumination joint editing results. The left-hand side shows the reconstructed image under the original scene lighting, and the right-hand side shows the relighted, material-modified images. Obviously, our editing results in different cases remain highly realistic.

TABLE V
ABLATION STUDY. TWO-BRANCH JOINT TRAINING CAN IMPROVE THE DECOMPOSITION ACCURACY.

	Basecolor			Specular			Roughness		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
w/o joint training	27.441	0.953	0.078	24.918	0.963	**0.058**	22.539	0.926	0.172
Ours (w/ joint training)	**28.860**	**0.962**	**0.072**	**26.258**	**0.966**	0.059	**27.882**	**0.972**	**0.069**

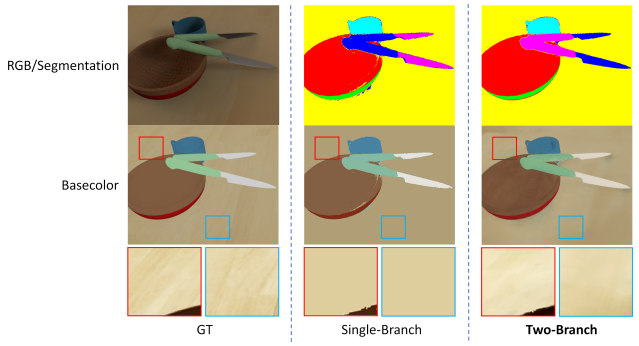

Fig. 12. Comparison of single-branch and two-branch frameworks. The contrast and saturation of local regions within the color boxes are adjusted for a better comparison. The single-branch model produces completely flattened basecolor and cannot reconstruct subtle variations within a single material. In contrast, our two-branch framework allows for small variations within the same type of material, resulting in better reconstruction of details in the scene.

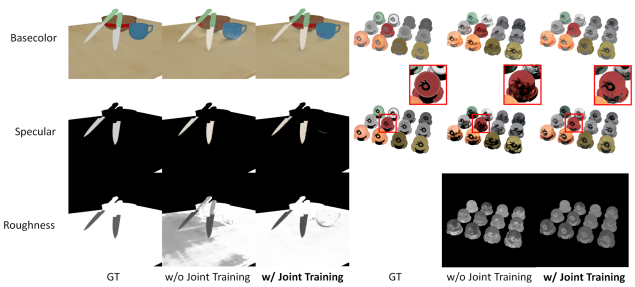

Fig. 13. During the two-branch joint training, VQ clustering compacts the latent distribution of material vectors, reducing decomposition noise.

to better reconstruct. This illustrates the essential role of the two-branch design in our method.

2) Ablation on Joint Training: To investigate the effectiveness of the joint training strategy of the two branches, we

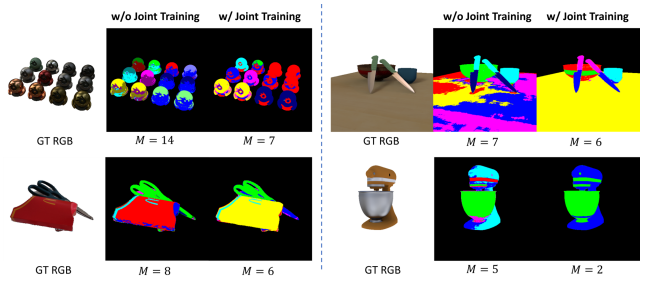

Fig. 14. When adopting joint training, VQ discretization produces accurate segmentation with less redundancy.

conducted an experiment by replacing the joint training with a separate training strategy. Specifically, we first trained the continuous branch independently of the discrete branch. Then, we fixed the continuous branch and trained the discrete branch. The quantitative results are reported in Table V, and the visualization results are shown in Figs. 13 and 14. Compared to the separate training strategy, the joint training strategy allows the continuous and discrete branches to benefit from each other during training. Specifically, our method with joint training can effectively suppress the noise in the predicted decomposed material, as shown in Fig. 13. Meanwhile, the material components produced by our method are clean and reasonable. In contrast, the method without joint training may produce confused materials, such as the knife handle and cup in the *kitchen* scene in Fig. 14.

3) VQ Clustering vs. Classical Clustering: To investigate the superiority of our VQ clustering, we conducted an experiment by replacing it with an intuitive clustering method. Specifically, we applied the *meanshift* clustering with three different bandwidths (0.5, 0.3, and 0.2) on our continuous branch for discrete material segmentation. As shown in Table VI and Fig. 15, both quantitative and qualitative results

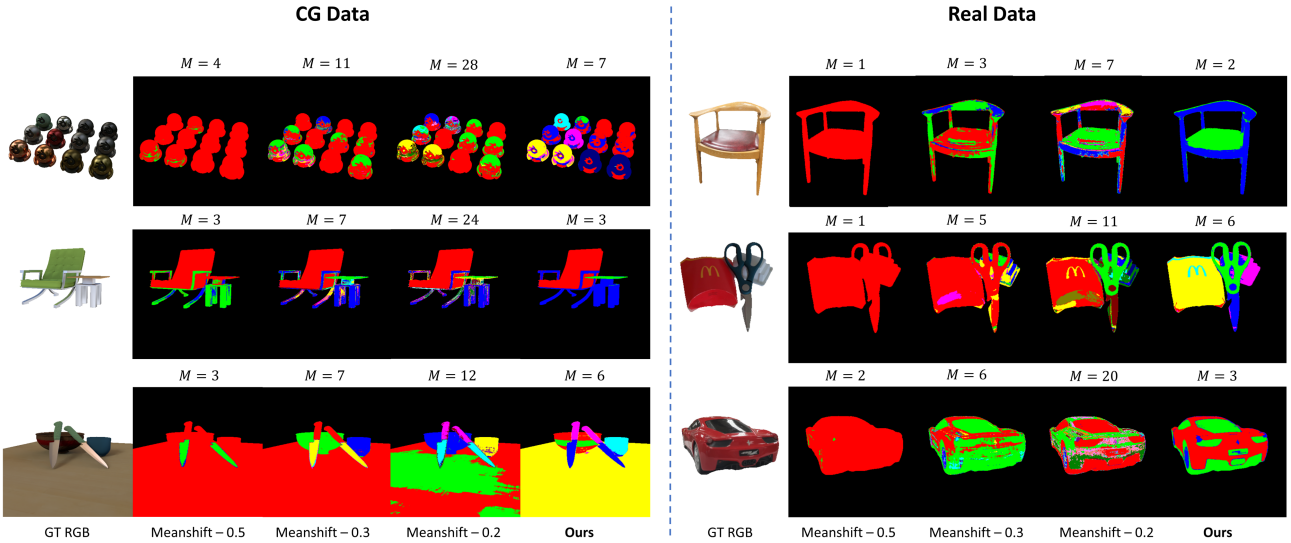

Fig. 15. Material segmentation results. M refers to the predicted number of scene materials. The left column shows the results on CG data, and the right column shows the results on real data. Evidently, our strategy produces the most accurate and least redundant segmentation on both CG and real data, demonstrating the superiority of our VQ clustering in discrete branch.

TABLE VI
MATERIAL SEGMENTATION RESULTS ON THE CG DATASET. THE VALUES OF THE THREE METRICS UNDER MICRO AVERAGE ARE THE SAME, SO WE ONLY PRESENT MICRO-F1 FOR COMPARISON.

	Material Clustering			
	Micro-F1↑	Macro-F1↑	Macro-P↑	Macro-R↑
Meanshift-0.2	0.747	0.350	0.357	0.384
Meanshift-0.3	0.629	0.268	0.296	0.303
Meanshift-0.5	0.503	0.127	0.146	0.173
Ours	**0.821**	**0.421**	**0.405**	**0.449**

demonstrate that the material segmentation produced by our VQ mechanism is more accurate and effective compared to the classical *meanshift* clustering method. For example, in the *metal-balls* scenario of Fig. 15, the *meanshift* clustering with different bandwidths fails to deduce reasonable and correct segmentation maps from only the continuous branch, while our VQ clustering in the discrete branch enables us to identify all the materials and infer an accurate material segmentation map. This advantage is attributed to learning clustering jointly with decomposition.

4) Ablation on Dropout-based Ranking: To illustrate the effectiveness of our dropout-based ranking strategy, we further compare our full method with the implementation without this strategy. Fig. 16 shows the visual comparison on discrete material segmentation. Here, in the implementation without the dropout-based ranking strategy, since the length of the VQ codebook cannot be automatically detected without our dropout-based ranking strategy, we set a constant $M = 15$ as the codebook length and show the segmentation map inferred from all codewords. Unlike the implementation without ranking strategy, our full method is able to rank codewords in importance and automatically determine the length of the VQ codebook, i.e., the number of materials. Therefore, our full method achieves to discard redundant codewords and generate compact material segmentation maps, which greatly facilitates

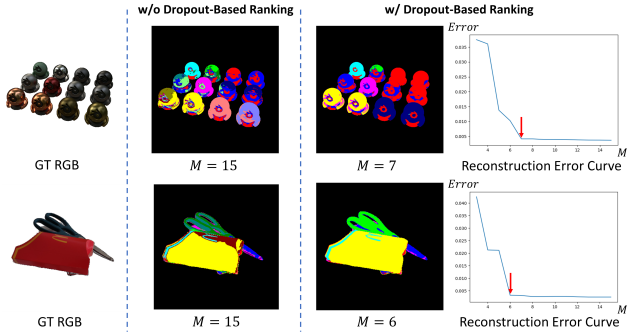

Fig. 16. Ablation results concerning the dropout-based ranking strategy, with the fourth column showing the reconstruction error curves. The horizontal axis of the curves represents the number of used materials M, while the vertical axis represents the reconstruction error. As depicted, the reconstruction error decreases as more materials are used. However, as shown by the results in the second column, using too many materials may result in redundant segmentation. Leveraging the dropout-based ranking strategy, we achieve a balance between reconstruction error and material redundancy, as demonstrated in the third column.

the subsequent material selection and editing process. Moreover, to prove the rationality of the length selection, we also show the reconstruction error curve as the number of materials changes in Fig. 16. As shown in the figure, the reconstruction error decreases as more materials are used. However, using too many materials may result in redundant segmentation. By introducing the dropout-based ranking strategy, our full method is able to select the most appropriate number of materials to balance reconstruction error and material redundancy.

V. CONCLUSION

In this paper, we propose VQ-NeRF, a VQ-based two-branch neural reflectance field for reflectance decomposition, material editing, and scene relighting. Unlike existing methods generate continuous material components, our VQ-NeRF in-

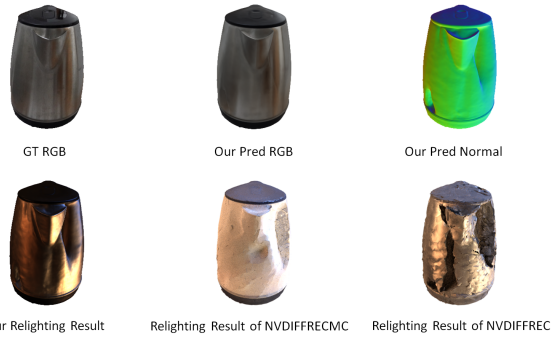

Fig. 17. Like other reflectance decomposition methods, inaccurate shapes produced by the geometry reconstruction method may result in relighting failures in our method.

troduce a discrete branch in addition to the continuous branch to produce discrete materials and deduce segmentation maps for facilitating material selection and editing. Meanwhile, we employ a dropout-based ranking strategy to eliminate material redundancy and automatically determine the number of scene materials. Moreover, we adopt a two-branch joint training strategy to encourage mutual benefit between the continuous and discrete branches and suppress the noise predicted in material components. Extensive experiments on both CG and real data demonstrate the superior performance of our VQ-NeRF in scene reconstruction, reflectance decomposition, material editing, and scene relighting tasks.

Limitations. In some cases, our method may fail to generate correct relighting or material editing results when the geometry reconstruction method (e.g., NeuS) cannot correctly model the geometric surface, as shown in Fig. 17. However, this limitation can potentially be overcome by developing more advanced neural implicit 3D representations or by jointly learning geometric reconstruction and reflectance decomposition. We leave these as future directions for our work.

REFERENCES

[1] Y. Zhang, J. Sun, X. He, H. Fu, R. Jia, and X. Zhou, "Modeling indirect illumination for inverse rendering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 18 643–18 652.

[2] K. Kang, M. Gu, C. Xie, X. Yang, H. Wu, and K. Zhou, "Neural reflectance capture in the view-illumination domain," *IEEE Transactions on Visualization and Computer Graphics*, 2021.

[3] X. Zhang, P. P. Srinivasan, B. Deng, P. Debevec, W. T. Freeman, and J. T. Barron, "Nerfactor: Neural factorization of shape and reflectance under an unknown illumination," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 6, pp. 1–18, 2021.

[4] M. Boss, R. Braun, V. Jampani, J. T. Barron, C. Liu, and H. Lensch, "Nerd: Neural reflectance decomposition from image collections," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 684–12 694.

[5] Z. Wang, T. Shen, J. Gao, S. Huang, J. Munkberg, J. Hasselgren, Z. Gojcic, W. Chen, and S. Fidler, "Neural fields meet explicit geometric representations for inverse rendering of urban scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 8370–8380.

[6] Y. M. Kim, S. Ryu, and I.-J. Kim, "Planar abstraction and inverse rendering of 3d indoor environments," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 6, pp. 2992–3006, 2019.

[7] W. Yang, G. Chen, C. Chen, Z. Chen, and K.-Y. K. Wong, "Ps-nerf: Neural inverse rendering for multi-view photometric stereo," in *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part I*. Springer, 2022, pp. 266–284.

[8] A. Meka, M. Maximov, M. Zollhoefer, A. Chatterjee, H.-P. Seidel, C. Richardt, and C. Theobalt, "Lime: Live intrinsic material estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6315–6324.

[9] Y. Sato, M. D. Wheeler, and K. Ikeuchi, "Object shape and reflectance modeling from observation," in *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, 1997, pp. 379–387.

[10] K. J. Dana, "Brdf/btf measurement device," in *Proceedings eighth ieee international conference on computer vision. ICCV 2001*, vol. 2. IEEE, 2001, pp. 460–466.

[11] X. Li, Y. Dong, P. Peers, and X. Tong, "Modeling surface appearance from a single photograph using self-augmented convolutional neural networks," *ACM Transactions on Graphics (ToG)*, vol. 36, no. 4, pp. 1–11, 2017.

[12] Z. Li, Z. Xu, R. Ramamoorthi, K. Sunkavalli, and M. Chandraker, "Learning to reconstruct shape and spatially-varying reflectance from a single image," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 6, pp. 1–11, 2018.

[13] M. Boss, V. Jampani, K. Kim, H. Lensch, and J. Kautz, "Two-shot spatially-varying brdf and shape estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 3982–3991.

[14] A. Joy and C. Poullis, "Multi-view gradient consistency for svbrdf estimation of complex scenes under natural illumination," *arXiv preprint arXiv:2202.13017*, 2022.

[15] Z. Li, L. Wang, M. Cheng, C. Pan, and J. Yang, "Multi-view inverse rendering for large-scale real-world indoor scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 12 499–12 509.

[16] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[17] S. Bi, Z. Xu, P. Srinivasan, B. Mildenhall, K. Sunkavalli, M. Hašan, Y. Hold-Geoffroy, D. Kriegman, and R. Ramamoorthi, "Neural reflectance fields for appearance acquisition," *arXiv preprint arXiv:2008.03824*, 2020.

[18] P. P. Srinivasan, B. Deng, X. Zhang, M. Tancik, B. Mildenhall, and J. T. Barron, "Nerv: Neural reflectance and visibility fields for relighting and view synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7495–7504.

[19] J. Guo, Y. Guo, J. Pan, and W. Lu, "Brdf analysis with directional statistics and its applications," *IEEE transactions on visualization and computer graphics*, vol. 26, no. 3, pp. 1476–1489, 2018.

[20] A. Van Den Oord, O. Vinyals *et al.*, "Neural discrete representation learning," *Advances in neural information processing systems*, vol. 30, 2017.

[21] J. Yu, X. Li, J. Y. Koh, H. Zhang, R. Pang, J. Qin, A. Ku, Y. Xu, J. Baldridge, and Y. Wu, "Vector-quantized image modeling with improved vqgan," *arXiv preprint arXiv:2110.04627*, 2021.

[22] H. Wu, Z. Wang, and K. Zhou, "Simultaneous localization and appearance estimation with a consumer rgb-d camera," *IEEE transactions on visualization and computer graphics*, vol. 22, no. 8, pp. 2012–2023, 2015.

[23] V. Deschaintre, M. Aittala, F. Durand, G. Drettakis, and A. Bousseau, "Single-image svbrdf capture with a rendering-aware deep network," *ACM Transactions on Graphics (ToG)*, vol. 37, no. 4, pp. 1–15, 2018.

[24] ——, "Flexible svbrdf capture with a multi-image deep network," in *Computer graphics forum*, vol. 38, no. 4. Wiley Online Library, 2019, pp. 1–13.

[25] J. L. Schonberger and J.-M. Frahm, "Structure-from-motion revisited," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4104–4113.

[26] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su, "Tensorf: Tensorial radiance fields," in *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXII*. Springer, 2022, pp. 333–350.

[27] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan, "Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5855–5864.

[28] K. Zhang, F. Luan, Q. Wang, K. Bala, and N. 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*, 2021, pp. 5453–5462.

[29] Z. Kuang, K. Olszewski, M. Chai, Z. Huang, P. Achlioptas, and S. Tulyakov, "Neroic: neural rendering of objects from online image collections," *ACM Transactions on Graphics (TOG)*, vol. 41, no. 4, pp. 1–12, 2022.

[30] K. Wang, S. Peng, X. Zhou, J. Yang, and G. Zhang, "Nerfcap: Human performance capture with dynamic neural radiance fields," *IEEE Transactions on Visualization and Computer Graphics*, 2022.

[31] M. Boss, V. Jampani, R. Braun, C. Liu, J. Barron, and H. Lensch, "Neural-pil: Neural pre-integrated lighting for reflectance decomposition," *Advances in Neural Information Processing Systems*, vol. 34, pp. 10 691–10 704, 2021.

[32] Y. Yao, J. Zhang, J. Liu, Y. Qu, T. Fang, D. McKinnon, Y. Tsin, and L. Quan, "Neilf: Neural incident light field for physically-based material estimation," in *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXI*. Springer, 2022, pp. 700–716.

[33] J. Munkberg, J. Hasselgren, T. Shen, J. Gao, W. Chen, A. Evans, T. Müller, and S. Fidler, "Extracting triangular 3d models, materials, and lighting from images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8280–8290.

[34] J. Hasselgren, N. Hofmann, and J. Munkberg, "Shape, light & material decomposition from images using monte carlo rendering and denoising," *arXiv preprint arXiv:2206.03380*, 2022.

[35] J. T. Kajiya, "The rendering equation," in *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, 1986, pp. 143–150.

[36] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang, "Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction," *arXiv preprint arXiv:2106.10689*, 2021.

[37] Y. Lu, Y. Chen, N. Ruozzi, and Y. Xiang, "Mean shift mask transformer for unseen object instance segmentation," *arXiv preprint arXiv:2211.11679*, 2022.

[38] A. Roy, A. Vaswani, A. Neelakantan, and N. Parmar, "Theory and experiments on vector quantized autoencoders," *arXiv preprint arXiv:1805.11063*, 2018.

[39] W. Ye, S. Chen, C. Bao, H. Bao, M. Pollefeys, Z. Cui, and G. Zhang, "Intrinsicnerf: Learning intrinsic neural radiance fields for editable novel view synthesis," *arXiv preprint arXiv:2210.00647*, 2022.

[40] Q. Wu, J. Tan, and K. Xu, "Palettenerf: Palette-based color editing for nerfs," *arXiv preprint arXiv:2212.12871*, 2022.

[41] Y. Zhang, T. Xu, J. Yu, Y. Ye, J. Wang, Y. Jing, J. Yu, and W. Yang, "Nemf: Inverse volume rendering with neural microflake field," *arXiv preprint arXiv:2304.00782*, 2023.

Supplementary Material for VQ-NeRF

In the following, we supplement the paper with additional results and details, including:

- More details about our geometry extraction.
- More details about our two-branch network.
- More information about our data collection.
- More experiments on our model, including ablation of loss functions and visualization of model optimization.
- A supplementary video.

I. Geometry Extraction

In our implementation, we employ the NeuS [1] for geometry reconstruction. In Neus, the geometry of a scene is modeled by an SDF f. Denote $\boldsymbol{r}(t) = \boldsymbol{o} + t\boldsymbol{d}$ as the spatial points sampled on the camera ray emitted from the origin $\boldsymbol{o}$ in the direction $\boldsymbol{d}$. The surface point $\boldsymbol{p}$ of ray $\boldsymbol{r}$ can be computed by:

$$\begin{cases} \boldsymbol{p}(\boldsymbol{r}) = \int_{t_n}^{t_f} T(t)\rho(f(\boldsymbol{r}(t)), t)\boldsymbol{r}(t)dt, \\ T(t) = \exp(-\int_{t_n}^{t} \rho(f(\boldsymbol{r}(u)), u)du), \end{cases} \tag{1}$$

where t_n and t_f represent the bounds of near and far sampling. $f(\boldsymbol{r}(t))$ represents the value of the SDF f at point $\boldsymbol{r}(t)$. $\rho(f(\boldsymbol{r}(t)), t)$ is the opaque density. Additionally, we follow the implementation of NeuS to compute the surface normal $\boldsymbol{N}(\boldsymbol{p})$:

$$\overline{\boldsymbol{N}}(\boldsymbol{p}) = \int_{t_n}^{t_f} T(t)\rho(f(\boldsymbol{r}(t)), t)\nabla_{\boldsymbol{r}(t)} f dt, \tag{2}$$

where $\nabla_{\boldsymbol{r}(t)} f$ is the gradient of f with respect to $\boldsymbol{r}(t)$. Then, we normalize $\overline{\boldsymbol{N}}(\boldsymbol{p})$ by $\boldsymbol{N}(\boldsymbol{p}) = \frac{\overline{\boldsymbol{N}}(\boldsymbol{p})}{||\overline{\boldsymbol{N}}(\boldsymbol{p})||}$ to obtain the final result.

II. Network Structure

1) Encoder Structure: The structure of the encoder f_e is illustrated in Fig. 1. The dark blue arrows indicate the fully-connected layers. The light blue squares represent the latent vectors. The numbers enclosed within the squares denote the vector dimensions. The input to the encoder is the surface point $\boldsymbol{p}$. $\boldsymbol{p}$ is first embedded by the positional encoding γ. Then, the embedded $\gamma(\boldsymbol{p})$ is processed through seven fully-connected layers to generate the latent material vector $\boldsymbol{z}$. A skip connection is established between the embedded inputs and the output vectors of the third layer.

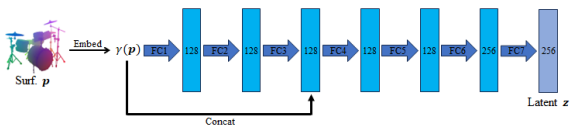

Fig. 1. The architecture of the encoder.

2) Decoder Structure: We employ similar architecture for the decoders f_d^c and f_d^d, as shown in Fig. 2. Both f_d^c and f_d^d consist of three MLPs for predicting different BRDF attributes (diffuse, specular, or roughness). In each MLP, the latent material vector $\boldsymbol{z}$ or the quantized codeword $\boldsymbol{z}_{vq}$ is processed through three fully-connected layers to predict the BRDF attribute with C channels. A skip connection is established between the inputs and the output vectors of the second layer. In the specific implementation, f_d^c uses the simplified Disney BRDF [2] for rigorous constraints. Therefore, its outputs require a linear transformation before rendering, as described in Sec. III-B of the paper.

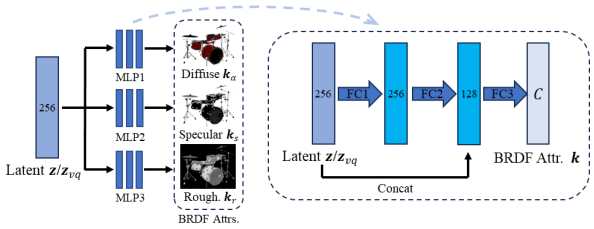

Fig. 2. The architecture of the decoders.

III. Data Collection

Due to some setting differences in source files, some scenes in the data released by NeRF and NeRFactor do not have ground truth of specular and roughness. To comprehensively evaluate our decomposition accuracy, we additionally collect three CG scenes containing the ground-truth of all BRDF attributes, and make our own dataset.

The geometry models, textures and illuminations used in these scenes are all collected online. Our data contains a variety of materials such as wood, metal, plastic, and fabric, allowing exhaustive evaluation of material decomposition. All of the scenes are lighted by HDR illumination maps, which are in the same format as the public data provided by NeRFactor. For rendering, we use the CYCLES engine of the Blender software, and employ the same camera poses used in the NeRF and NeRFactor datasets. We keep the same rendering settings as NeRFactor and use their released scripts to generate our data.

TABLE I
Ablation on Chromaticity Loss.

	Material Clustering			
	Micro-F1↑	Macro-F1↑	Macro-P↑	Macro-R↑
w/o L_{chr}	0.755	0.374	0.376	0.406
Ours (w/ L_{chr})	**0.821**	**0.421**	**0.405**	**0.449**

TABLE II
ABLATION STUDY. ALTHOUGH THE LAMBERTIAN LOSS SLIGHTLY REDUCES THE ACCURACY OF BASECOLOR AND ROUGHNESS, IT SIGNIFICANTLY ENHANCES THE DECOMPOSITION OF THE SPECULAR ATTRIBUTE.

	Basecolor			Specular			Roughness		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
w/o L_{lam}	**29.914**	**0.967**	**0.059**	13.676	0.639	0.272	**30.386**	**0.979**	**0.043**
Ours (w/ L_{lam})	28.860	0.962	0.072	**26.258**	**0.966**	**0.059**	27.882	0.972	0.069

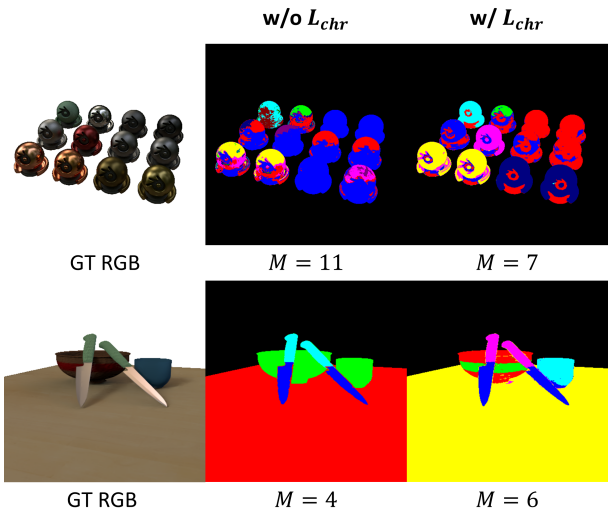

Fig. 3. Ablation on chromaticity loss. The introduction of chromaticity loss improves the segmentation accuracy of surface points with dark colors.

IV. SUPPLEMENTARY EXPERIMENTS

A. Ablation on Loss Functions

To demonstrate the necessity of adopting L_{chr}, L_{lam} and L_{sm} in model optimization, we conduct more experiments to evaluate their impact on material decomposition and segmentation.

1) Ablation on L_{chr}: Our model works on data captured under arbitrary lighting, and in some scenes, the luminance of the appearance color varies greatly across different surface points. Therefore, those surface points with dark colors only have a small loss when computing the reconstruction loss in RGB space. This presents challenges for the model optimization on those dark points. Such problem is severe in our VQ segmentation, leading to confusion between different materials with dark colors. To address this issue, we introduce a chromaticity loss L_{chr} in the discrete branch. L_{chr} computes the reconstruction loss in chromaticity space, where all colors are normalized to the same lightness 1. Therefore, the computed loss is independent of the luminance of apperance color, so surface points with dark colors can be strongly constrained, leading to better convergence.

We compare our method to a model trained without L_{chr}. Table I and Fig. 3 show the quantitative and qualitative results. Apparently, the chromaticity loss improves the segmentation accuracy on surface points with dark colors, such as the matte parts of red balls and black-green balls in the *metal-balls* scene in Fig. 3.

2) Ablation on L_{lam}: Due to the ambiguity of inverse rendering, different combinations of BRDF attributes can render to the same appearance. A typical example is that a specular

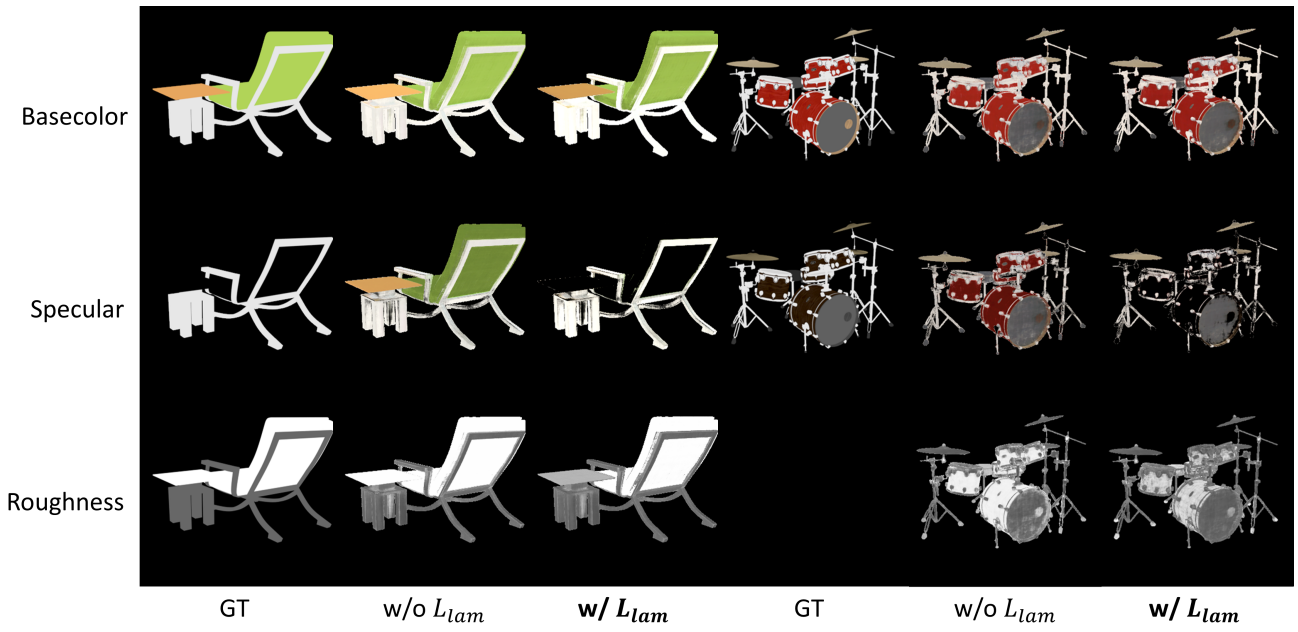

Fig. 4. Ablation on Lambertian loss. The incorporation of Lambertian loss significantly improves the decomposition accuracy of specular attributes.

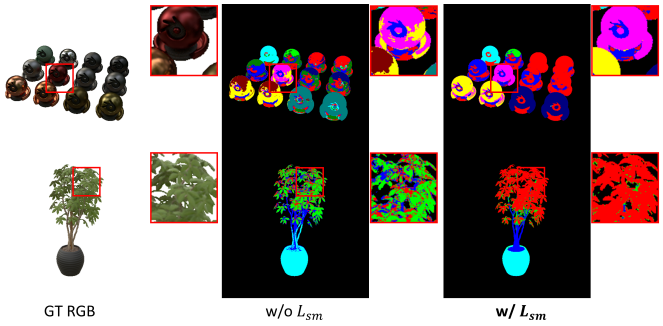

Fig. 5. With the help of the smooth loss, our method produces concise segmentation results, which is beneficial for convenient editing.

material with high roughness can exhibit a similar appearance to a diffuse material. To introduce more constraints to resolve this ambiguity, we propose a Lambertian loss that predicts surface points with high roughness to have low specular, which is consistent with real material relations.

We compare our method to a model trained without L_{lam}. Table II and Fig. 4 give the quantitative and qualitative results. As shown, although the incorporation of L_{lam} slightly reduces the decomposition accuracy of the basecolor and roughness, it significantly improves the decomposition quality of the specular attribute.

TABLE III
ABLATION ON SMOOTH LOSS.

	Material Clustering			
	Micro-F1↑	Macro-F1↑	Macro-P↑	Macro-R↑
w/o L_{chr}	0.779	0.372	0.348	0.411
Ours (w/ L_{chr})	**0.821**	**0.421**	**0.405**	**0.449**

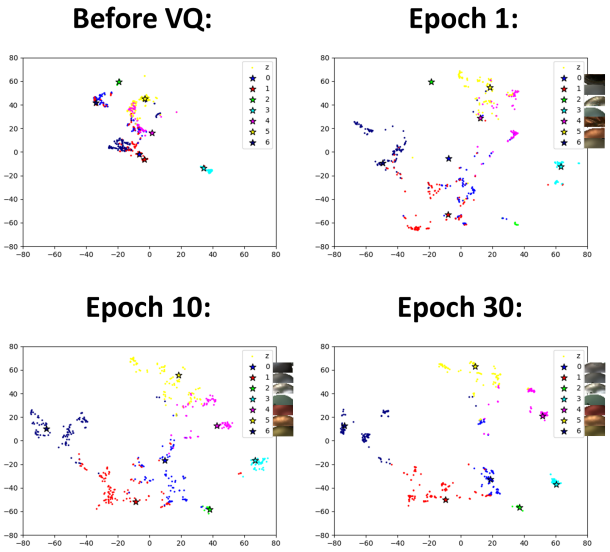

Fig. 6. Visualization of the model optimization using t-SNE. Dots represent continuous vectors (cluster samples), stars represent discrete VQ codewords (cluster centers). As the training progresses, different materials are gradually pulled apart and materials of the same type are compacted together. This demonstrates the effectiveness of our optimization strategy, bringing benefits such as noise suppression and segmentation enhancement.

3) Ablation on L_{sm}: To make our discrete branch produce smooth segmentation results, we introduce a weighted smooth loss L_{sm}. To demonstrate its effectiveness, we compare our method to a model trained without L_{sm}. Table III and Fig. 5 present the quantitative and qualitative results. Obviously, the smooth loss helps our model produce more concise segmentations, which is beneficial for convenient material editing.

B. Visualization of Model Optimization

We use t-SNE to visualize the model optimization for the *metal-balls* scene. The corresponding figure is shown in Fig. 6. Dots represent the continuous material vectors, and stars represent the corresponding VQ codewords. Material categories are mixed before the optimization. As training progresses, individual materials are gradually pulled apart into isolated clusters. Since the latent distribution of individual materials is compacted in each cluster, the decomposition noise is suppressed accordingly. On the other hand, with the simultaneous adjustment of the material vectors (cluster samples), the VQ codewords (cluster centers) can converge to more appropriate positions, thereby improving the segmentation accuracy.

REFERENCES

[1] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang, "Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction," *arXiv preprint arXiv:2106.10689*, 2021.

[2] Y. Yao, J. Zhang, J. Liu, Y. Qu, T. Fang, D. McKinnon, Y. Tsin, and L. Quan, "Neilf: Neural incident light field for physically-based material estimation," in *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXI*. Springer, 2022, pp. 700–716.