Image and Vision Computing 164 (2025) 105793

Contents lists available at ScienceDirect

Image and Vision Computing

journal homepage: www.elsevier.com/locate/imavis

Simultaneous acquisition of geometry and material for translucent objects

Chenhao Li [a,*], Trung Thanh Ngo [b], Hajime Nagahara [a]

[a] Osaka University, 2-8 Yamadaoka, Suita, Japan
[b] Hanoi University of Science and Technology, 1 Dai Co Viet, Hai Ba Trung, Hanoi, Viet Nam

ARTICLE INFO

MSC:
41A05
41A10
65D05
65D17

Keywords:
Inverse rendering
Translucent objects
Shape from single image

ABSTRACT

Reconstructing the geometry and material properties of translucent objects from images is a challenging problem due to the complex light propagation of translucent media and the inherent ambiguity of inverse rendering. Therefore, previous works often make the assumption that the objects are opaque or use a simplified model to describe translucent objects, which significantly affects the reconstruction quality and limits the downstream tasks such as relighting or material editing. We present a novel framework that tackles this challenge through a combination of physically grounded and data-driven strategies. At the core of our approach is a hybrid rendering supervision scheme that fuses a differentiable physical renderer with a learned neural renderer to guide reconstruction. To further enhance supervision, we introduce an augmented loss tailored to the neural renderer. Our system takes as input a flash/no-flash image pair, enabling it to disambiguate complex light propagation that happens inside translucent objects. We train our model on a large-scale synthetic dataset of 117 K scenes and evaluate across both synthetic benchmarks and real-world captures. To mitigate the domain gap between synthetic and real data, we contribute a new real-world dataset with ground-truth surface normals and fine-tune our model accordingly. Extensive experiments validate the robustness and accuracy of our method across diverse scenarios.

1. Introduction

Understanding and correctly reconstructing how translucent materials interact with light is a fundamental challenge in computer vision and graphics. Unlike opaque surfaces, translucent objects exhibit complex light transport phenomena, notably *Subsurface Scattering* (SSS), where photons penetrate the surface, scatter internally, and exit at various locations. This leads to a rich variety of appearance effects that are hard to model and reconstruct from captured images. Accurately recovering the intrinsic components – geometry, material properties, and illumination – from images of translucent objects remains a largely unsolved and ill-posed problem due to the inherently entangled nature of these factors.

Recent advances in *inverse rendering* have yielded impressive results for opaque and even glossy surfaces [1–4]. However, these methods assume surface-only reflection models and fail to account for the SSS present in translucent objects. Meanwhile, specialized works targeting translucent materials have either isolated SSS while ignoring surface reflectance [5,6], or relied on simplified representations such as BSSRDFs [7,8] that are insufficient for general translucent objects with complex light transportations.

To address these limitations, we propose a novel inverse rendering framework capable of jointly estimating surface geometry, spatially-varying reflectance, homogeneous subsurface scattering parameters, and environmental illumination for general translucent objects. Our system takes as input a flash/no-flash image pair from a single viewpoint, a configuration inspired by recent bidirectional reflectance acquisition techniques [2,10], which helps to disambiguate surface and subsurface scattering in image formation.

Key to our approach is the combined use of two renderers: a physically-based renderer that models direct surface reflection, and a neural renderer trained to approximate the complex indirect scattering pathways characteristic of SSS. Both renderers are differentiable so that we can train them end-to-end. To further enhance learning, we introduce an *augmented loss* that supervises the neural renderer by computing the SSS parameters perturbed re-rendering images.

To support learning and evaluation, we construct a large-scale synthetic dataset of over 117,000 translucent scenes, featuring human-designed 3D objects with physically based BRDF and SSS properties under environment maps. Additionally, we curate a real-world benchmark comprising 89 translucent objects with ground-truth normal maps to evaluate generalization. Since synthetic-to-real transfer remains a major hurdle, we perform fine-tuning on the real dataset, mitigating the domain gap and significantly improving real-world performance.

Our contributions can be summarized as follows:

* Corresponding author.
E-mail address: lch@is.ids.osaka-u.ac.jp (C. Li).

https://doi.org/10.1016/j.imavis.2025.105793
Received 26 December 2024; Received in revised form 12 July 2025; Accepted 13 October 2025

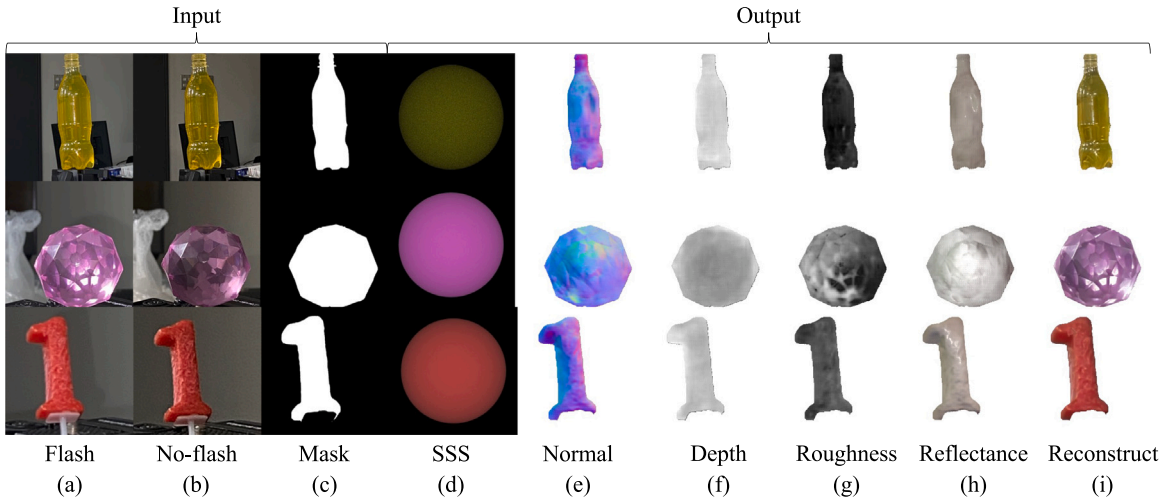

Fig. 1. Inverse rendering results of real-world translucent objects. Our model takes (a) a flash image, (b) a no-flash image, and (c) a mask as input. Then, it decomposes them into (d) homogeneous subsurface scattering parameters (denoted as SSS in the figure), (e) surface normal, (f) depth, and (g) spatially-varying roughness. We use the predicted parameters to re-render the image that only considers (h) the surface reflectance, and (i) both surface reflectance and subsurface scattering. Note that (d) subsurface scattering only contains 7 parameters (3 for extinction coefficient σ_t, 3 for volumetric albedo α, 1 for phase function parameter g), we use these parameters to render a sphere using Mitsuba [9] for visualization. Brighter areas in the depth map represent a greater distance, and brighter areas in the roughness map represent a rougher surface.

- We tackle the inverse rendering of translucent objects by jointly estimating shape, spatially-varying surface reflectance, SSS parameters, and illumination from a single-view flash/no-flash pair.
- We propose a hybrid rendering architecture combining physically accurate surface modeling with a learned neural renderer for SSS.
- We introduce a novel augmented loss strategy to improve the supervision of SSS effects.
- We build a large-scale synthetic dataset and a real-world benchmark to train and validate our model.
- We demonstrate state-of-the-art performance on both synthetic and real translucent objects, supported by fine-tuning to address domain shift.

This work is an extended version of Li et al. [11]. Compared to the original manuscript, the extensions are (1) an additional real-world translucent dataset that contains 89 images with ground truth surface normals and (2) a fine-tuning step on the real-world dataset to alleviate the domain gap problem. (3) An ablation study using different encoder backbones. (4) Adding more comparisons between the latest works.

2. Related work

2.1. Inverse rendering of surface reflectance

So far several efforts have been devoted into estimating depth [12–14], BRDF [4,10,15–18], and illumination [19–21] separately.

With the rise of deep learning, simultaneous parameter estimation has attracted people's attention. Li et al. [22] introduced the first single-view method to estimate shape, SVBRDF, and illumination using a cascaded network and an in-network rendering layer. This concept has been extended to more complex scenarios, such as indoor scene inverse rendering using a Residual Appearance Renderer [23] and spatially-varying lighting [24–26]. Li et al. [27] further extended this to complex materials like metal or mirrors. Boss et al. [2] addressed the ambiguity under saturated highlights with a two-shot setup. Sang and Chandraker [28] combined shape and SVBRDF estimation with relighting using both physically-based and neural renderers. Deschaintre et al. [29] and Li et al. [30] incorporated polarization into shape and SVBRDF estimation. Wu et al. [31] trained models unsupervised, leveraging rotational symmetry. Lichy et al. [32] achieved high-resolution shape and material reconstruction using a recursive neural architecture. Recent works have introduced diffusion models into inverse rendering to better handle ambiguity and multimodal decomposition. RGBX [33] and IntrinsicAnything [34] learn generative priors for intrinsic properties such as albedo, roughness, and lighting, enabling both decomposition and synthesis. Neural LightRig [35] leverages diffusion-based relighting to improve normal and BRDF estimation, while Material Anything [36] offers end-to-end generation of PBR materials in UV space. Diffusion Posterior Illumination [37] incorporates illumination priors into differentiable rendering, enabling ambiguity-aware inverse solutions. Our model can be considered as an extension of these methods by integrating SSS into the estimation process.

2.2. Inverse rendering of subsurface scattering

Reconstructing translucent materials with subsurface scattering (SSS) is challenging due to the complex, multipath nature of light transport. Early methods [5,38–41] relied on Monte Carlo volume rendering and analysis-by-synthesis optimization, but suffered from high computational cost and convergence issues. Later approaches such as Che et al. [6] adopted neural networks to predict homogeneous scattering parameters, enabling faster initialization. However, these methods often neglect geometry, lighting, or surface reflectance, and assume pure SSS. Recent works address these gaps: Neural Relighting [42] combines radiance transfer learning with geometry refinement for better relighting; Neural SSS [43] proposes a compact neural BSSRDF model for heterogeneous scattering; NeuralTO [44] improves geometry reconstruction and view synthesis by modifying radiance fields for translucent media. Others focus on appearance editing [45] or material transfer from images to 3D models [46]. Our method builds upon these by jointly estimating geometry, illumination, surface reflectance, and SSS from a single-view flash/no-flash pair, handling both surface and subsurface light propagation in a unified framework.

2.3. Differentiable rendering

Differentiable renderers are widely used in the inverse rendering domain for applications such as the reconstruction of human face [3,47], indoor scenes [23,24], buildings [48,49], and single objects [2,22,28]. However, most of these methods rely on physically-based renderers that account only for direct illumination. This limitation hinders

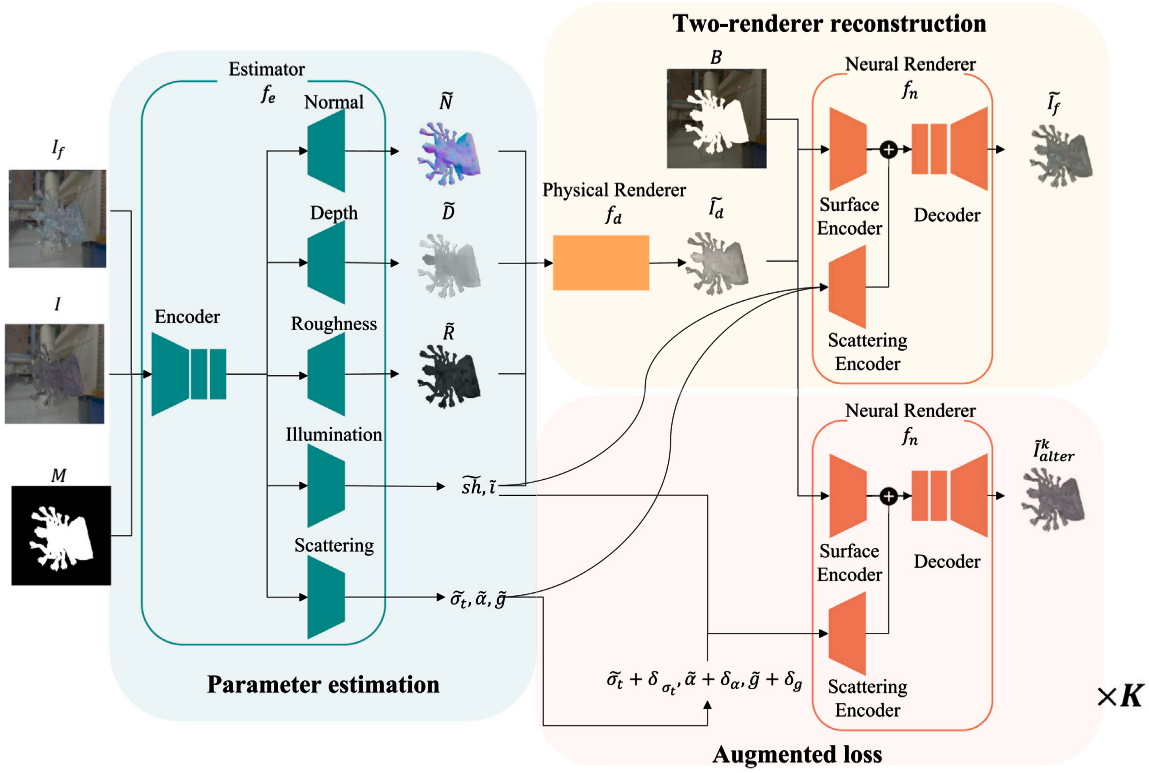

Fig. 2. Overview of the proposed model. We use different colors to indicate different functions: green for the estimator, orange for the physically-based renderer, and red for the neural renderer. We have K augmented loss modules and only show one in the figure. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

their ability to produce high-quality images, as they cannot simulate effects like soft shadows, inter-reflection, and subsurface scattering. Recent studies have introduced general-purpose differentiable renderers [9,50–52] that incorporate indirect illumination. Nevertheless, these Monte Carlo path tracing approaches are computationally intensive and require significant memory, especially with high resolutions. On the other hand, alternative approaches [22,23,28,53–55] have integrated neural networks with direct illumination renderers to enable global illumination and scene editing. These methods inspired us to develop a renderer-neural network architecture specifically for subsurface scattering tasks.

2.4. Scene editing

Scene editing is a vast area of research, with recent advancements driven largely by deep learning. Numerous studies have focused on relighting [3,49,56–59], material editing [60,61] and object manipulation [62–65]. Our work pioneers the editing of scattering parameters for translucent objects based on observations from just two images.

3. Methods

In this section, we introduce our proposed inverse rendering framework. Section 3.1 details our input and output modeling. Section 3.2 describes the neural network used for parameter estimation. In Section 3.3, we introduce the two-renderer structure. Section 3.4 discusses an augmented loss to enhance the supervision of the neural renderer. Section 3.5 presents the loss function. Finally, we detail the fine-tuning step in Section 3.6.

3.1. Problem setup

Scene representation To represent geometry, we use a depth map D for the rough shape and a normal map N for local details. For the surface, we adopt a microfacet BSDF model proposed by Walter et al. [66], where a roughness map R is used to define the surface roughness. Homogeneous SSS is modeled using three terms: an extinction coefficient σ_t controlling optical density, a volumetric albedo α determining photon scattering or absorption probability during a volume event, and a Henyey–Greenstein phase function [67] parameter g indicating the scattering direction (forward when $g > 0$, backward when $g < 0$, and isotropic when $g = 0$). Additionally, we estimate spherical harmonics sh to aid the model and predict a flashlight intensity i accounting for varying flashlight intensities across devices.

Model design Inspired by Aittala et al. [10] and Boss et al. [2], we use a flash and no-flash image setup, taking advantage of the different visibility of translucent objects under different lighting intensities. For example, if we put a bright light on the back of our finger, we can clearly see the color of blood. This property facilitates scattering parameter estimation and better disentanglement of surface reflectance and SSS.

Given a translucent object with an unknown shape, material, and illumination, we aim to estimate these parameters simultaneously and enable material editing by manipulating the estimated parameters. Fig. 2 provides an overview of our model. Our inputs are three images: a flash image $I_f \in \mathbb{R}^{3\times256\times256}$, a no-flash image $I \in \mathbb{R}^{3\times256\times256}$, and a binary mask $M \in \mathbb{R}^{256\times256}$. The estimated parameters for each scene are:

- A depth map $D \in \mathbb{R}^{256\times256}$ and a normal map $N \in \mathbb{R}^{3\times256\times256}$ to represent the shape.

Fig. 3. Examples of the proposed synthetic dataset: we use the ground truth shape and illumination to visualize SSS parameters.

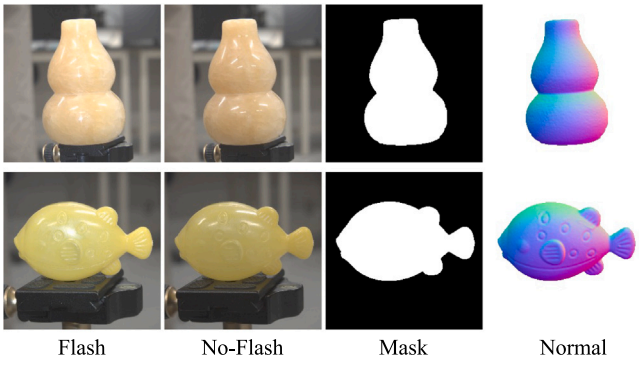

Fig. 4. Examples of the proposed real-world dataset.

- A roughness map $R \in \mathbb{R}^{256\times256}$ used in the microfacet BSDF model.
- The spherical harmonics $sh \in \mathbb{R}^{3\times9}$ and a flashlight intensity $i \in \mathbb{R}^1$ to represent illumination.
- The extinction coefficient $\sigma_t \in \mathbb{R}^3$, volumetric albedo $\alpha \in \mathbb{R}^3$, and Henyey–Greenstein phase function parameter $g \in \mathbb{R}^1$.

3.2. Parameter estimation

We employ a deep convolutional neural network with a single encoder and multiple heads for parameter estimation, treating shape, material, and illumination estimation as a multi-task learning problem. The encoder extracts features from the input images while each head estimates the respective parameters. So, given a flash image I_f, no-flash image I, and a binary mask M, the estimated physical parameters are:

$$\tilde{D}, \tilde{N}, \tilde{R}, \tilde{sh}, \tilde{i}, \tilde{\sigma_t}, \tilde{\alpha}, \tilde{g} = f_e(I, I_f, M), \tag{1}$$

where f_e denotes the estimator. $\tilde{D}, \tilde{N}, \tilde{R}, \tilde{sh}, \tilde{i}, \tilde{\sigma_t}, \tilde{\alpha}, \tilde{g}$ represent the estimated depth, normal, roughness, spherical harmonics coefficient, flashlight intensity, extinction coefficient, volumetric albedo, and Henyey–Greenstein phase function parameter (see Fig. 3).

3.3. Physical renderer and neural renderer

Reconstruction loss is commonly used to supervise network training. In the inverse rendering field, reconstruction means re-rendering the scene with the estimated parameters. Additionally, the renderer must be differentiable to propagate the reconstruction loss gradient to the estimator.

While general-purpose differentiable renderers like Mitsuba [9] are available, they pose memory and speed issues on consumer-grade GPUs, especially at high resolutions. The reason for that is the computational cost of path tracing is much larger than that of standard neural works. For example, differentially rendering a single translucent object image with 256 × 256 resolution and 64 samples per pixel (spp) requires more than 5 s and 20 GB memory on an RTX3090 GPU. Moreover, the rendering time and memory will increase linearly when spp increases. Additionally, rendering SSS requires full 3D shape estimation (*e.g.*, a 3D mesh), which is difficult from a single viewpoint. That is why we only estimate a normal and a depth map as our geometry representation, which is sufficient for rendering surface reflectance but not for SSS.

Alternative approaches [2,31] might be using a differentiable renderer that only accounts for direct illumination. This sacrifices some reconstruction quality but dramatically improves efficiency. However, such a method cannot be applied to a translucent object because SSS depends on multiple bounces of light inside the object.

Inspired by the recent advances in image-to-image translation [68, 69], researchers have shown the successful application of neural networks for adding indirect illumination [22,53], photorealistic effect [23, 48], or relighting [28,54]. Inspired by them, we propose a two-step rendering pipeline to mimic the rendering process of a general-purpose differentiable renderer. The first step is a physically-based rendering module considering only direct illumination:

$$\tilde{I_d} = f_d(\tilde{D}, \tilde{N}, \tilde{R}, \tilde{i}, \tilde{sh}), \tag{2}$$

where f_d is a physically-based renderer that follow the implementation of Sang and Chandraker [28], and it has no trainable parameters. $\tilde{I_d}$ is the re-rendered image that only considers the surface reflectance. The second step is a neural renderer f_n to create the SSS effect:

$$\tilde{I_f} = f_n(\tilde{I_d}, \tilde{sh}, \tilde{i}, \tilde{\sigma_t}, \tilde{\alpha}, \tilde{g}, B), \tag{3}$$

where $\tilde{I_f}$ is the re-rendered flash image. f_n is the proposed neural renderer, and it consists of 3 parts (See Fig. 2 for reference): a Surface encoder, a Scattering encoder, and a decoder. The Surface encoder maps the estimated surface reflectance image $\tilde{I_d}$ into a feature map. In addition to the surface reflectance image, we also input the background image B (masked-out version of I_f) to the Surface encoder to provide high-frequency illumination information. The Scattering encoder consists of a few upsampling layers. It maps $\tilde{sh}, \tilde{i}, \tilde{\sigma_t}, \tilde{\alpha}, \tilde{g}$ to a feature map. The decoder consists of some Resnet blocks [70] and upsampling layers. Our neural renderer's task can be considered a conditional image-to-image translation, where the condition is the SSS parameters.

The advantage of the two-renderer design is that it is naturally differentiable. At the same time, the training cost is acceptable. In addition, the physically-based renderer can provide physical hints to the neural renderer. Because we separate the reconstruction of surface reflectance and SSS explicitly, the ambiguity problem can be alleviated.

3.4. Augmented loss

In this subsection, we present an augmented loss to address the hidden information problem by using multiple altered images to supervise the proposed neural renderer. As discussed in 3.3, we use a two-renderer structure to compute the reconstruction images, improving parameter estimation. However, a well-known problem of reconstruction loss in deep learning is that neural networks learn to "hide" information within them [71], causing the neural renderer to ignore the estimated SSS parameters and only reconstruct the input image based on the hidden information. If so, the reconstruction loss cannot give correct gradients and thus fails to guide the training of the estimator.

Table 1
MAE results on 17,140 test scenes. The best three results in each column are highlighted with gold, silver, and bronze backgrounds.

	Geometry		BSDF	Illumination		SSS		
	N	D	R	sh	i	σ_t	α	g
Baseline	.0918	.0705	.0811	.1083	.0912	.1670	.1061	.1762
2R	.0916	.0697	.0811	.1064	.0908	.1675	.1057	.1777
2R-AUG	.0913	.0699	.0807	.1105	.0893	.1619	.1040	.1703
Che et al. [6]	–	–	–	–	–	.1828	.1115	.2123
Ours (ResNet)	.0894	.0646	.0769	.0989	.0804	.1590	.1002	.1655
Ours(Swin2)	.0951	.0794	.0781	.0901	.0825	.1594	.0968	.1689
Ours(ConvNext2)	.0874	.0634	.0868	.1037	.0891	.1781	.1107	.1878

This undermines the effectiveness of reconstruction loss in guiding the estimator training.

To address this problem, we enhance the supervision of the neural renderer with an augmented loss. Specifically, after the parameters are estimated, we edit the estimated SSS parameters and let the Neural renderer reconstruct image based on the edited SSS parameters. We also render K altered images I^k_{alter} as their ground truth labels to train the neural renderer. The altered images have the same parameters as the original flash image, except the SSS parameters. Specifically, I_f and I^k_{alter} share the same shape, surface reflectance, and illumination, but different extinction coefficient, phase function, and volumetric albedo. The edited SSS parameters are randomly sampled from the same distribution as the original ones:

$$\tilde{I}^k_{alter} = f_n(\tilde{I}_d, \tilde{sh}, \tilde{i}, \tilde{\sigma}_t + \delta_{\sigma_t}, \tilde{\alpha} + \delta_\alpha, \tilde{g} + \delta_g, B), \tag{4}$$

where $\delta_{\sigma_t}, \delta_\alpha, \delta_g$ are the differences between target SSS parameters (subsurface scattering parameters of I^k_{alter}) and the estimated ones, $\tilde{I}^k_{alter}$ is the reconstructed altered images. This design offers several benefits. First, since the input image is not the same as the image to be reconstructed, it becomes meaningless for the neural network to "hide" information from the original input image. Second, the variety of input parameters and output images makes the neural renderer more sensitive to changes in SSS parameters. These factors help the neural renderer guide the estimator more effectively, leading to a more accurate estimation of SSS parameters. Considering the time and computing resources required to render the training images, we set $K = 3$ in practice.

3.5. Loss functions

The proposed model is fully supervised and trained end to end, and we compute the loss between the estimated parameters their ground truths:

$$\begin{aligned} L = {} & L_D + L_N + L_R + L_{sh} + L_i \\ & + L_{\sigma_t} + L_\alpha + L_g + L_{I_f} + \sum_K L^k_{alter}, \end{aligned} \tag{5}$$

where $L_D, L_N, L_R, L_{I_f}, L^k_{alter}$ stand for the $L1$ loss between the estimated depth, normal, roughness, flash image, altered images and their ground truth ones. $L_{sh}, L_i, L_{\sigma_t}, L_\alpha, L_g$ stands for the $L2$ loss between the estimated spherical harmonics, flashlight intensity, extinction coefficient, volumetric albedo, Henyey–Greenstein phase function parameter and their ground truth ones.

3.6. Real-world data fine-tuning

As briefly mentioned before, a well-known problem of training on the synthetic data and testing on the real-world data is the domain gap problem. We take two solutions to deal with the domain gap problem in this work. The first one is to make the data distribution of synthetic and real data as close as possible. Specifically, we design the rendering of synthetic images using realistic geometry, material, and illumination, which will be discussed in Section 4.1.1. The second one, which will be discussed in this section, is an additional fine-tuning step on the real-world data.

However, the inherent problem of training on real-world data is the lack of ground truth labels. The problem is even escalated with translucent objects. Because most measurement methods of geometry and material are designed for opaque objects. Directly applying these methods to translucent objects usually fails. Nevertheless, we use an anti-translucency spray to change all translucent objects to opaque and use a 3D scanner to measure the ground-truth geometries (details can be found in Section 4.1.2). Finally, we use the constructed real-world dataset with ground truth surface normals to finetune our model. During the finetuning, only the Normal head is continually trained, and the other parts of the Estimator, and the Neural Renderer, are frozen. Due to the lack of so many ground truth parameters, we can only compute the reconstruction loss L_{I_f} and the surface normal loss L_N. As a consequence, the loss function during the finetuning step is:

$$L_{ft} = L_N + LI_f \tag{6}$$

4. Experiments

We introduce our dataset in Section 4.1. In Section 4.2, we compare our model with an existing inverse scattering work [6]. We conduct an ablation study and report the quantitative and qualitative results in Section 4.3. In Section 4.4 we show the results of SSS parameter editing application. Finally, in Section 4.5, we show the improvement of the model after the finetuning step.

4.1. Datasets

4.1.1. Synthetic data

Collecting a large number of real-world translucent objects with measurements of shape, surface reflectance, and SSS parameters is time-consuming. However, training data is essential for deep neural networks. To address this, we created a large-scale synthetic dataset through photorealistic rendering. However, as briefly mentioned earlier, the difference between the distribution of synthetic data and real-world data introduces the domain gap problem. We carefully designed the rendered images to mitigate the domain difference between the synthetic and real data. This section details the preparation of assets for rendering, including 3D objects, BSDF maps, subsurface scattering parameters, and illumination.

3D objects Previous works [2,22,28] have synthesized 3D objects using the Domain Randomized method by assembling simple shapes like spheres, cylinders, and cones. We collected human-created 3D objects from ShapeNet [72] and some other public resources for greater shape complexity and diversity. Some ShapeNet objects had flipped surface normals, resulting in black pixels when intersecting with light. We used a script to delete these objects, retaining 5847 3D objects. All objects were scaled to fit within a 50 cm cube, placed at the origin,

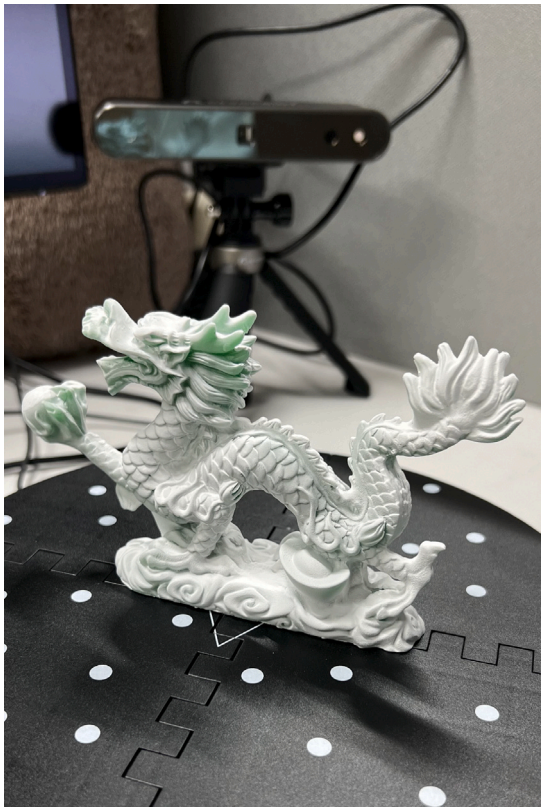

Fig. 5. We paint the object with an anti-translucency spray and use a 3D scanner to capture the GT geometry of translucent objects.

and randomly rotated, scaled, and translated during rendering. We used 5000 objects for training and 857 for testing.

Roughness and bump maps To create meaningful surface reflectance patterns, we collected surface reflectance maps from open-source websites, each containing a bump map and a roughness map. Using Blender's "smart UV mapping" function [73], we applied these maps to the objects. The bump map was used to modify the surface normals for a more realistic appearance. All roughness and bump maps were randomly resized before applying them to the object. We collected 2745 surface reflectance maps, using 2470 for training and 275 for testing.

IoR To reduce the ambiguity problem in inverse rendering, we set the IoR to a constant value (1.5046), which is typical for many common materials like glass (1.5046), amber (1.55), and polyethylene (1.49).

Subsurface scattering parameters Following the previous work [6], we randomly sampled the subsurface scattering parameters from a uniform distribution with extinction coefficient $\sigma_t \in [0, 32]$, volumetric albedo $\alpha \in [0.3, 0.95]$, and Henyey–Greenstein phase function parameter $g \in [0, 0.9]$.

Illumination To replicate complex real-world lighting conditions, we used environment maps from the Laval Indoor HDR dataset [19], which consists of 2357 high-resolution indoor panoramas. During rendering, pitch and roll were fixed, with only the yaw axis rotated. We computed a 3×9 Spherical Harmonics coefficient for each environment map to supervise our model. We used 1500 environment maps for training and 857 for testing. Initially, we used a point light to simulate the flashlight but observed severe noise with smooth objects. Instead, we used a small sphere area light with a 10 cm radius placed 10 cm behind the camera. To account for varying flashlight intensities in different devices, we used random radiance values between 35 to 75 W $m^{-2}sr^{-1}$.

Rendering For each scene, we used Mitsuba2 [9] to render five images: flash image, no-flash image, and three altered images. Each scene included a randomly selected object, normal map, roughness map, environment map, and subsurface scattering parameters. The camera was placed 70 cm away from the origin on the positive z-axis, looking at the origin. For the flash image, a small area light source simulated the camera flashlight. For the no-flash image, the area light was removed, and the camera's look-at direction was slightly altered to simulate camera shake. The subsurface scattering parameters were edited for the three altered images. Additionally, we rendered ground truth depth, normal, roughness, and binary masks for intermediate supervision. We generated a total of 100,000 training scenes and 17,140 test scenes.

4.1.2. Real data

For a more comprehensive test of our model, we also constructed a real-world dataset consisting of several common translucent objects. We also measured the ground truth surface normal for each scene. However, even if we only focus on measuring the surface normals of translucent objects, it is still not easy. Existing works [74,75] measure the geometry using 3D scanners and register the measured geometry and the image to the same coordinate system using an image-to-geometry alignment method [76]. However, neither the 3D geometry measuring nor the geometry registration are designed for translucent objects. We solved the geometry measuring problem by using an anti-translucency spray (See Fig. 5) and measured the geometry using a REVEPOINT POP 2 scanner. For the geometry registration problem. According to the existing method [76], the object position is optimized by computing the loss of the image space between the re-rendered one and the camera-captured one. However, during the re-rendering, the material and illumination are unknown. Therefore, they used a standard illumination and material and proposed a novel mutual information loss that is robust to the change of illumination and material. Although the proposed mutual information may work for opaque objects, translucent objects have more ambiguities. Moreover, their rendering system is not differentiable, and they used a quadric approximation, which can be slow and inaccurate. Instead, we used Mitsuba [77] for the rendering and computed the loss on the silhouette space to optimize the position. Finally, we obtained 80 training scenes and 9 test scenes. Each scene contains a flash photo, a no-flash photo, a manually created binary mask, and a ground truth surface normal. We show a few examples of the constructed real-world data in Fig. 4.

4.2. State-of-the-art comparison

4.2.1. Material estimation comparison

To the best of our knowledge, we are the first to address the inverse rendering problem for translucent objects that involve both surface reflectance and SSS. Comparing our method to those focusing solely on **pure surface reflectance**, such as [2,22], is not easy. Although these methods predict parameters like surface roughness, they rely on the coloration affected by "diffuse albedo", a parameter of the BRDF. In contrast, our scene representation models surface reflection and refraction using BSDF and SSS using the Radiative Transport Equation (RTE). This difference means that in our model, the coloration is influenced by the volumetric albedo, extinction coefficient, and phase function, making it impossible to train pure surface reflectance methods on our dataset. Therefore, we chose to compare our model with a **pure SSS** method proposed by Che et al. [6]. Their method requires an edge map as an additional input for the neural network. Following their procedure, we generated edge maps for our dataset and trained their model using the same hyperparameters as our method. We report the Mean Absolute Error (MAE) in Table 1, and provide a visual comparison in Fig. 6. Due to the lack of ground truth parameter references, we only present results for synthetic data. Our observations indicate that their method struggles to estimate reasonable SSS parameters because of the highly ambiguous scene representation.

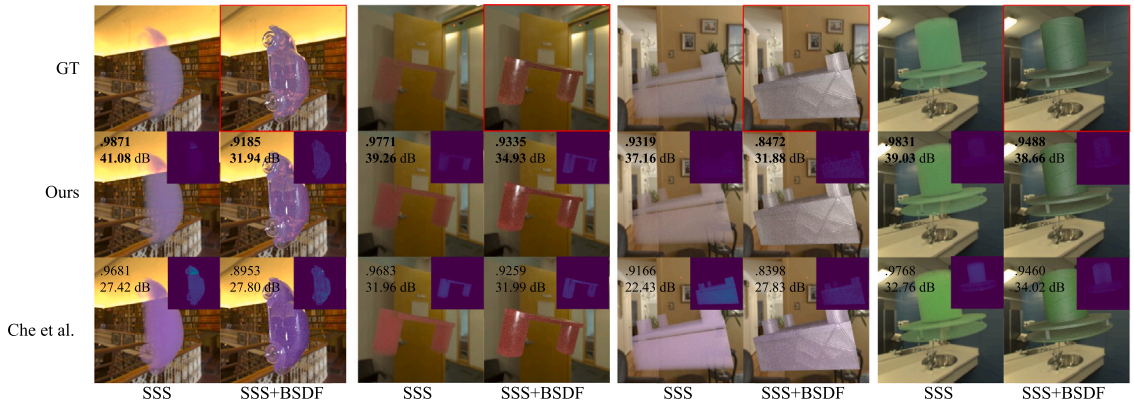

Fig. 6. Visual comparison with [6]. We use two visualizations, which are "SSS" and "SSS+BSDF" to show the parameter estimation results. "SSS" is rendered by the GT illumination, shape, and predicted SSS. "SSS+BSDF" is rendered by the GT illumination, shape, BSDF, and predicted SSS. Images in red rectangles are input images. Error maps are in the upper right corner. SSIM and PSNR values are in the upper left corner.

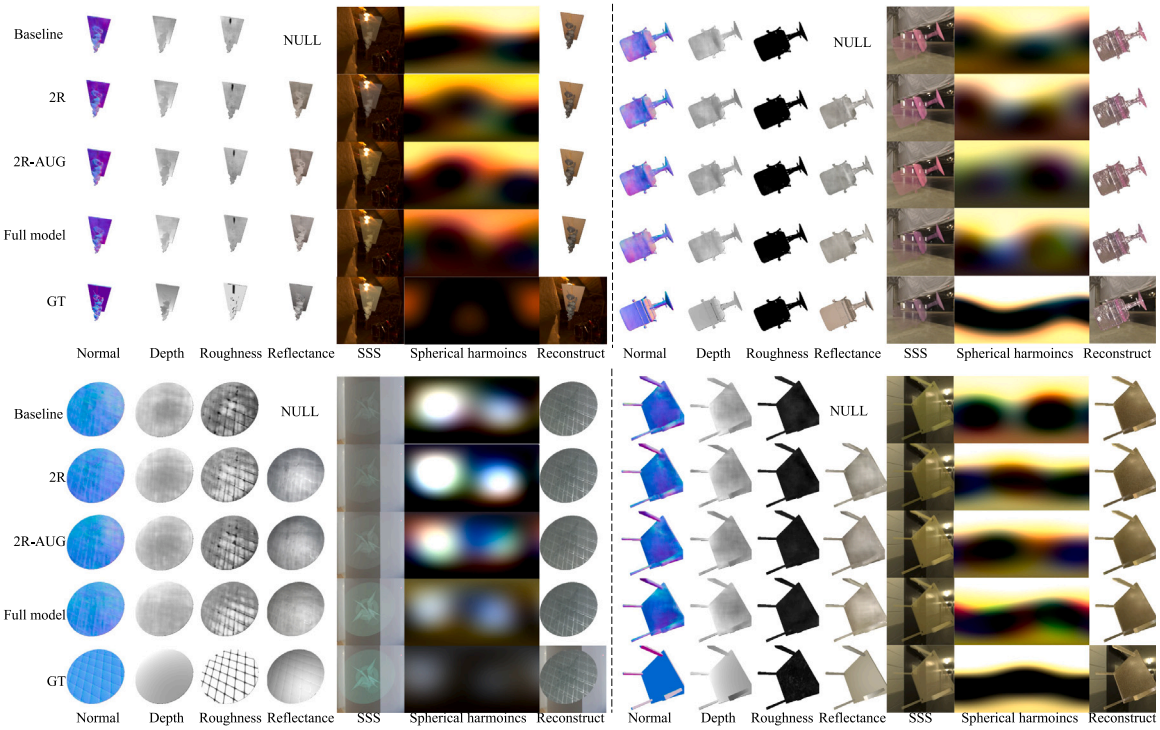

Fig. 7. Visual results of ablation study on the synthetic dataset. For visualization, we use the ground truth shape and illumination to render the image of the estimated SubSurface Scattering parameters (denoted as SSS in the figure).

Fig. 8. SSS parameter editing results. The 3rd row is the input images. We randomly edit the scattering parameters before inputting them into the neural renderer. The 2nd row is the estimated images, and we show their ground truth images in the 1st row.

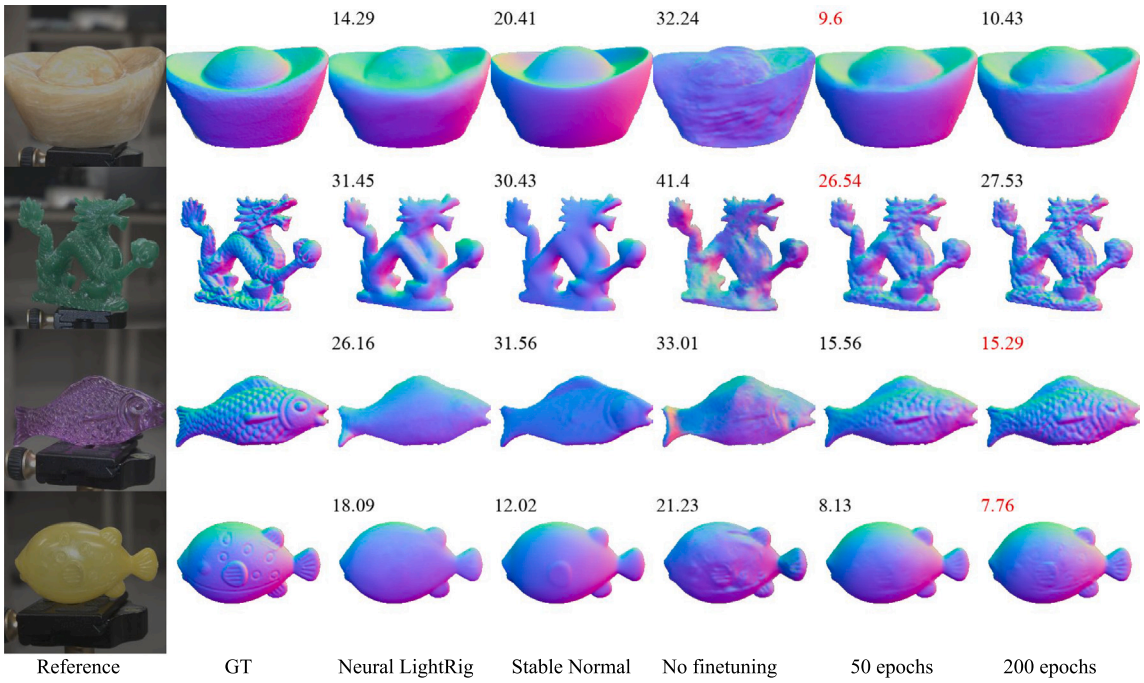

Fig. 9. Finetuning results of the surface normal estimation and comparison between the existing works. Numbers on the top left stand for the mean angular error between estimated normals and GT normals. The best results are highlighted in red. Finetuning results come from Ours(ConvNext) model.

4.2.2. Normal estimation comparison

Although we cannot compare the material estimation results with the pure surface reflection models due to the different material modeling, the estimated surface normals can still be compared. Thus, we conduct a surface normal estimation comparison to the existing methods. We selected two representative works, Stable Normal [78] and Neural LightRig [35], that can estimate high-fidelity normals to be our competitors. Results are demonstrated in Fig. 9. It is observed that although our method cannot defeat the latest work for the normal estimation, it outperforms these works after fine-tuning. This result is consistent with our expectations. These diffusion-based methods are usually pre-trained on a large-scale dataset; thus, they usually have a better generalization ability to real-world scenes. In contrast, our synthetic dataset is relatively small; therefore, it requires an additional fine-tuning stage. Nevertheless, it is observed that after the fine-tuning, our method has a better performance due to the consideration of complex light propagation for translucent objects.

4.3. Ablation study

4.3.1. Model design

We conducted ablation studies to evaluate the impact of each component of our proposed model. We began with a Baseline model, which uses the same estimator as our Full model but only reconstructs the input scene using a neural network. Essentially, this baseline model functions as an autoencoder. Next, we divided the reconstruction process into two steps: a physically-based renderer to reconstruct the surface reflectance and a neural renderer to create multi-bounce illumination and the SSS effect. We refer to this as the "2R" model. We then introduced the augmented loss to the "2R" model, labeling it "2R-AUG". Finally, we achieved the Full model by incorporating the two-shot setting into "2R-AUG". The MAE results for the synthetic data from all experiments are reported in Table 1. The "2R" model outperformed the Baseline model in most metrics, particularly in illumination accuracy, confirming that explicitly separating surface reflectance and SSS reduces ambiguity. Although "2R" and "2R-AUG" performed similarly in geometry, illumination, and surface reflectance, the augmented loss improved the SSS parameters' results. The comparison between "2R-AUG" and the Full model demonstrates that the two-shot setting further reduces ambiguity. Fig. 7 shows visual comparisons, highlighting that the Full model's SSS images closely resemble the ground truth, while other models are less stable. Additionally, the Full model's estimation of environment illumination is more accurate due to the two-shot setting. To demonstrate the flexibility of our model, we tested it on several common real-world translucent objects, as illustrated in Fig. 1. The results show that our model can estimate reasonable SSS parameters.

4.3.2. Backbone search

To investigate the impact of the encoder backbone on our model's performance, we conducted extensive experiments using three representative architectures: ResNet-52 [70], ConvNeXt V2-Base [79], and Swin Transformer V2-Base [80]. These backbones were chosen due to their popularity and effectiveness in various vision tasks. Importantly, we selected architectures with comparable parameter counts to ensure a fair comparison of feature extraction capabilities.

As summarized in Table 1, we found that all three backbones yielded similar overall performance in our framework. Notably, ConvNeXt V2 achieved marginally better results in normal and depth estimation tasks. We hypothesize that this improvement arises from its hybrid design, which blends the inductive biases of convolutional networks with architectural modernizations inspired by vision transformers. These characteristics allow ConvNeXt V2 to retain strong generalization even when trained on limited data, while benefiting from a more flexible and expressive architecture compared to traditional CNNs.

On the other hand, the Swin Transformer V2 backbone did not show significant improvement over the CNN-based backbones. A likely reason is that vision transformers, including Swin Transformer, are known to be data-hungry and often require large-scale pretraining to fully leverage their potential. In contrast, convolutional networks possess strong spatial inductive biases, which make them more suitable for tasks with relatively limited training data. Our findings are consistent with conclusions drawn in prior work on vision transformers [81,82], where transformers outperform CNNs primarily when trained with extensive datasets and longer training schedules.

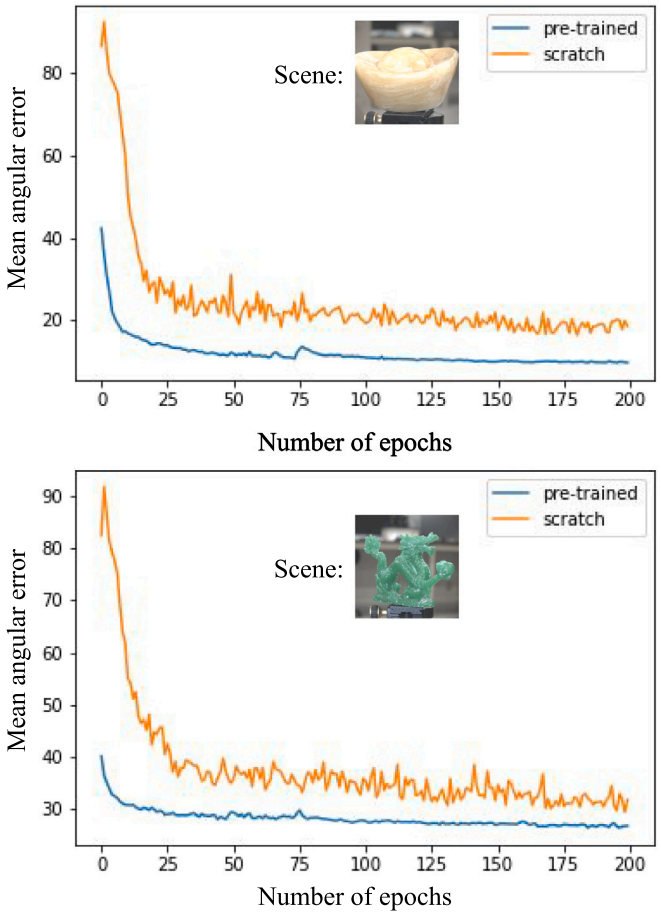

Fig. 10. Comparison of with or without pre-training on the synthetic data. The vertical axis represents the mean angular error between the estimated surface normals and their ground truths. The horizontal axis represents the number of finetuning epochs.

4.4. Material editing

Given a translucent object with an unknown shape, illumination, and material, we show that the proposed inverse rendering framework can edit translucent objects based on the given SSS parameters. Fig. 8 shows the editing results.

4.5. Finetuning analysis

We show the experimental results of the finetuning step in this section. After training on the synthetic data, we continue to finetune the model on the real-world data for 200 epochs. Fig. 9 demonstrates the contribution of the proposed finetuning step. As shown in the figure, without finetuning, the model tends to estimate a flat surface, and the problem is alleviated after several finetuning steps. In addition, we conducted an ablation study by comparing the training with or without the pretraining on the synthetic data. We show the results in Fig. 10. From the figure, we can observe that the training is much more stable compared to training from scratches.

5. Conclusions and limitations

In this paper, we made the first attempt to jointly estimate shape, spatially varying surface reflectance, homogeneous SSS, and illumination from a flash and no-flash image pair. The proposed two-shot, two-renderer, and augmented loss reduced the ambiguity of inverse rendering and improved the parameter estimation. We also constructed a large-scale synthetic dataset of fully labeled translucent objects and a real-world dataset with ground truth surface normals. In addition, we demonstrated that our pipeline is also capable of SSS parameter editing. Finally, we demonstrated that the model performance can be further improved and robust to the domain gap by adding a finetuning step using real-world data.

The proposed method still has some limitations. First, to reduce the ambiguity problem, we set the IoR to be constant and do not estimate it. However, IoR affects light transmission through object boundaries and thus influences the SSS. For some materials with large or small IoR, our model may not work. Second, our model does not support relighting and novel-view synthesis. Unlike pure surface reconstruction models, the estimated normals can be easily applied to physically-based renderers for relighting or novel-view synthesis. Rendering translucent objects requires a full 3D estimation (e.g., mesh), including the backside information. However, estimating the complete geometry is difficult for the single-view reconstruction method. Solving these challenges can be good for future work.

CRediT authorship contribution statement

Chenhao Li: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Data curation, Conceptualization. **Trung Thanh Ngo:** Writing – review & editing, Supervision, Methodology, Conceptualization. **Hajime Nagahara:** Writing – review & editing, Supervision, Conceptualization.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work, the authors used ChatGPT (GPT 4o) in order to enhance the clarity and readability of part of the text. After using this tool/service, the authors reviewed and edited the content as needed and took full responsibility for the publication's content.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This paper is partially supported by JSPS KAKENHI 23H05490.

Data availability

Data will be made available on request.

References

[1] J.T. Barron, J. Malik, Shape, illumination, and reflectance from shading, IEEE Trans. Pattern Anal. Mach. Intell. 37 (8) (2014) 1670–1687.
[2] M. Boss, V. Jampani, K. Kim, H. Lensch, 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.
[3] S. Sengupta, A. Kanazawa, C.D. Castillo, D.W. Jacobs, Sfsnet: Learning shape, reflectance and illuminance of facesin the wild', in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 6296–6305.
[4] Z. Li, K. Sunkavalli, M. Chandraker, Materials for masses: SVBRDF acquisition with a single mobile phone image, in: Proceedings of the European Conference on Computer Vision, ECCV, 2018, pp. 72–87.
[5] I. Gkioulekas, S. Zhao, K. Bala, T. Zickler, A. Levin, Inverse volume rendering with material dictionaries, ACM Trans. Graph. 32 (6) (2013) 1–13.
[6] C. Che, F. Luan, S. Zhao, K. Bala, I. Gkioulekas, Towards learning-based inverse subsurface scattering, in: 2020 IEEE International Conference on Computational Photography, ICCP, IEEE, 2020, pp. 1–12.

[7] X. Deng, F. Luan, B. Walter, K. Bala, S. Marschner, Reconstructing translucent objects using differentiable rendering, in: ACM SIGGRAPH 2022 Conference Proceedings, 2022, pp. 1–10.

[8] B. Dong, K.D. Moore, W. Zhang, P. Peers, Scattering parameters and surface normals from homogeneous translucent materials using photometric stereo, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 2291–2298.

[9] M. Nimier-David, D. Vicini, T. Zeltner, W. Jakob, Mitsuba 2: A retargetable forward and inverse renderer, ACM Trans. Graph. 38 (6) (2019) 1–17.

[10] M. Aittala, T. Weyrich, J. Lehtinen, et al., Two-shot SVBRDF capture for stationary materials, ACM Trans. Graph. 34 (4) (2015) 1–13.

[11] C. Li, T.T. Ngo, H. Nagahara, Inverse rendering of translucent objects using physical and neural renderers, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 12510–12520.

[12] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, V. Koltun, Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer, 2019, arXiv preprint arXiv:1907.01341.

[13] H. Fu, M. Gong, C. Wang, K. Batmanghelich, D. Tao, Deep ordinal regression network for monocular depth estimation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 2002–2011.

[14] R. Li, K. Xian, C. Shen, Z. Cao, H. Lu, L. Hang, Deep attention-based classification network for robust depth prediction, in: Asian Conference on Computer Vision, Springer, 2018, pp. 663–678.

[15] M. Aittala, T. Aila, J. Lehtinen, Reflectance modeling by neural texture synthesis, ACM Trans. Graph. (ToG) 35 (4) (2016) 1–13.

[16] X. Li, Y. Dong, P. Peers, X. Tong, Modeling surface appearance from a single photograph using self-augmented convolutional neural networks, ACM Trans. Graph. (ToG) 36 (4) (2017) 1–11.

[17] V. Deschaintre, M. Aittala, F. Durand, G. Drettakis, A. Bousseau, Single-image svbrdf capture with a rendering-aware deep network, ACM Trans. Graph. (ToG) 37 (4) (2018) 1–15.

[18] A. Meka, M. Maximov, M. Zollhoefer, A. Chatterjee, H.-P. Seidel, C. Richardt, C. Theobalt, Lime: Live intrinsic material estimation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 6315–6324.

[19] M.-A. Gardner, K. Sunkavalli, E. Yumer, X. Shen, E. Gambaretto, C. Gagné, J.-F. Lalonde, Learning to predict indoor illumination from a single image, 2017, arXiv preprint arXiv:1704.00090.

[20] S. Georgoulis, K. Rematas, T. Ritschel, M. Fritz, T. Tuytelaars, L. Van Gool, What is around the camera? in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 5170–5178.

[21] Y. Hold-Geoffroy, K. Sunkavalli, S. Hadap, E. Gambaretto, J.-F. Lalonde, Deep outdoor illumination estimation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 7312–7321.

[22] Z. Li, Z. Xu, R. Ramamoorthi, K. Sunkavalli, M. Chandraker, Learning to reconstruct shape and spatially-varying reflectance from a single image, ACM Trans. Graph. 37 (6) (2018) 1–11.

[23] S. Sengupta, J. Gu, K. Kim, G. Liu, D.W. Jacobs, J. Kautz, Neural inverse rendering of an indoor scene from a single image, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 8598–8607.

[24] Z. Li, M. Shafiei, R. Ramamoorthi, K. Sunkavalli, M. Chandraker, Inverse rendering for complex indoor scenes: Shape, spatially-varying lighting and svbrdf from a single image, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 2475–2484.

[25] Z. Wang, J. Philion, S. Fidler, J. Kautz, Learning indoor inverse rendering with 3D spatially-varying lighting, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, ICCV, 2021, pp. 12538–12547.

[26] R. Zhu, Z. Li, J. Matai, F. Porikli, M. Chandraker, IRISformer: Dense vision transformers for single-image inverse rendering in indoor scenes, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 2822–2831.

[27] Z. Li, L. Wang, X. Huang, C. Pan, J. Yang, PhyIR: Physics-based inverse rendering for panoramic indoor images, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 12713–12723.

[28] S. Sang, M. Chandraker, Single-shot neural relighting and svbrdf estimation, in: European Conference on Computer Vision, Springer, 2020, pp. 85–101.

[29] V. Deschaintre, Y. Lin, A. Ghosh, Deep polarization imaging for 3D shape and SVBRDF acquisition, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 15567–15576.

[30] C. Li, T. Ono, T. Uemori, H. Mihara, A. Gatto, H. Nagahara, Y. Moriuchi, NeISF: Neural incident Stokes field for geometry and material estimation, 2023, arXiv preprint arXiv:2311.13187.

[31] S. Wu, A. Makadia, J. Wu, N. Snavely, R. Tucker, A. Kanazawa, De-rendering the world's revolutionary artefacts, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 6338–6347.

[32] D. Lichy, J. Wu, S. Sengupta, D.W. Jacobs, Shape and material capture at home, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 6123–6133.

[33] Z. Zeng, V. Deschaintre, I. Georgiev, Y. Hold-Geoffroy, Y. Hu, F. Luan, L.-Q. Yan, M. Hašan, RGB↔X: Image decomposition and synthesis using material- and lighting-aware diffusion models, in: ACM SIGGRAPH 2024 Conference Papers, SIGGRAPH '24, Association for Computing Machinery, New York, NY, USA, 2024, http://dx.doi.org/10.1145/3641519.3657445.

[34] X. Chen, S. Peng, D. Yang, Y. Liu, B. Pan, C. Lv, X. Zhou, Intrinsicanything: Learning diffusion priors for inverse rendering under unknown illumination, in: European Conference on Computer Vision, Springer, 2024, pp. 450–467.

[35] Z. He, T. Wang, X. Huang, X. Pan, Z. Liu, Neural LightRig: Unlocking accurate object normal and material estimation with multi-light diffusion, in: Proceedings of the Computer Vision and Pattern Recognition Conference, 2025, pp. 26514–26524.

[36] X. Huang, T. Wang, Z. Liu, Q. Wang, Material anything: Generating materials for any 3d object via diffusion, in: Proceedings of the Computer Vision and Pattern Recognition Conference, 2025, pp. 26556–26565.

[37] L. Lyu, A. Tewari, M. Habermann, S. Saito, M. Zollhöfer, T. Leimkühler, C. Theobalt, Diffusion posterior illumination for ambiguity-aware inverse rendering, ACM Trans. Graph. 42 (6) (2023) 1–14.

[38] I. Gkioulekas, A. Levin, T. Zickler, An evaluation of computational imaging techniques for heterogeneous inverse scattering, in: European Conference on Computer Vision, Springer, 2016, pp. 685–701.

[39] A. Levis, Y.Y. Schechner, A. Aides, A.B. Davis, Airborne three-dimensional cloud tomography, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 3379–3387.

[40] P. Khungurn, D. Schroeder, S. Zhao, K. Bala, S. Marschner, Matching real fabrics with micro-appearance models, ACM Trans. Graph. 35 (1) (2015) 1–1.

[41] S. Zhao, L. Wu, F. Durand, R. Ramamoorthi, Downsampling scattering parameters for rendering anisotropic media, ACM Trans. Graph. 35 (6) (2016) 1–11.

[42] S. Zhu, S. Saito, A. Bozic, C. Aliaga, T. Darrell, C. Lassner, Neural relighting with subsurface scattering by learning the radiance transfer gradient, 2023, arXiv preprint arXiv:2306.09322.

[43] T. Tg, D.M. Tran, H.W. Jensen, R. Ramamoorthi, J.R. Frisvad, Neural SSS: Lightweight object appearance representation, in: Computer Graphics Forum, vol. 43, no. 4, Wiley Online Library, 2024, e15158.

[44] Y. Cai, J. Qiu, Z. Li, B. Ren, NeuralTO: Neural reconstruction and view synthesis of translucent objects, ACM Trans. Graph. 43 (4) (2024) 1–14.

[45] D. Lanza, B. Masia, A. Jarabo, Navigating the manifold of translucent appearance, in: Computer Graphics Forum, vol. 43, no. 2, Wiley Online Library, 2024, e15035.

[46] X. Wang, Y. Cheng, Z. Fan, K. Xu, Learning to transfer heterogeneous translucent materials from a 2D image to 3D models, in: Proceedings of the 32nd ACM International Conference on Multimedia, 2024, pp. 3440–3448.

[47] Y. Wen, W. Liu, B. Raj, R. Singh, Self-supervised 3D face reconstruction via conditional estimation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 13289–13298.

[48] M. Meshry, D.B. Goldman, S. Khamis, H. Hoppe, R. Pandey, N. Snavely, R. Martin-Brualla, Neural rerendering in the wild, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 6878–6887.

[49] Y. Yu, W.A. Smith, InverseRenderNet: Learning single image inverse rendering, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 3155–3164.

[50] T.-M. Li, M. Aittala, F. Durand, J. Lehtinen, Differentiable monte carlo ray tracing through edge sampling, ACM Trans. Graph. 37 (6) (2018) 1–11.

[51] C. Zhang, L. Wu, C. Zheng, I. Gkioulekas, R. Ramamoorthi, S. Zhao, A differential theory of radiative transfer, ACM Trans. Graph. 38 (6) (2019) 227:1–227:16.

[52] C. Zhang, B. Miller, K. Yan, I. Gkioulekas, S. Zhao, Path-space differentiable rendering, ACM Trans. Graph. 39 (4) (2020) 143:1–143:19.

[53] O. Nalbach, E. Arabadzhiyska, D. Mehta, H.P. Seidel, T. Ritschel, Deep shading: convolutional neural networks for screen space shading, in: Computer Graphics Forum, vol. 36, Wiley Online Library, 2017, pp. 65–78.

[54] Y. Yu, A. Meka, M. Elgharib, H.-P. Seidel, C. Theobalt, W.A. Smith, Self-supervised outdoor scene relighting, in: European Conference on Computer Vision, Springer, 2020, pp. 84–101.

[55] C. Zhang, Z. Yu, S. Zhao, Path-space differentiable rendering of participating media, ACM Trans. Graph. 40 (4) (2021) 1–15.

[56] R. Pandey, S.O. Escolano, C. Legendre, C. Haene, S. Bouaziz, C. Rhemann, P. Debevec, S. Fanello, Total relighting: learning to relight portraits for background replacement, ACM Trans. Graph. 40 (4) (2021) 1–21.

[57] A. Liu, S. Ginosar, T. Zhou, A.A. Efros, N. Snavely, Learning to factorize and relight a city, in: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16, Springer, 2020, pp. 544–561.

[58] Z. Xu, K. Sunkavalli, S. Hadap, R. Ramamoorthi, Deep image-based relighting from optimal sparse samples, ACM Trans. Graph. (ToG) 37 (4) (2018) 1–13.

[59] H. Zhou, S. Hadap, K. Sunkavalli, D.W. Jacobs, Deep single-image portrait relighting, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 7194–7202.

[60] G. Liu, D. Ceylan, E. Yumer, J. Yang, J.-M. Lien, Material editing using a physically based rendering network, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2261–2269.

[61] J. Shi, Y. Dong, H. Su, S.X. Yu, Learning non-lambertian object intrinsics across shapenet categories, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1685–1694.
[62] S. Yao, T.M.H. Hsu, J.-Y. Zhu, J. Wu, A. Torralba, W.T. Freeman, J.B. Tenenbaum, 3D-aware scene manipulation via inverse graphics, 2018, arXiv preprint arXiv:1808.09351.
[63] J. Wu, J.B. Tenenbaum, P. Kohli, Neural scene de-rendering, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 699–707.
[64] T.D. Kulkarni, W. Whitney, P. Kohli, J.B. Tenenbaum, Deep convolutional inverse graphics network, 2015, arXiv preprint arXiv:1503.03167.
[65] Y. Zhang, W. Chen, H. Ling, J. Gao, Y. Zhang, A. Torralba, S. Fidler, Image gans meet differentiable rendering for inverse graphics and interpretable 3d neural rendering, 2020, arXiv preprint arXiv:2010.09125.
[66] B. Walter, S.R. Marschner, H. Li, K.E. Torrance, Microfacet models for refraction through rough surfaces, Render. Tech. 2007 (2007) 18th.
[67] L.G. Henyey, J.L. Greenstein, Diffuse radiation in the galaxy, Astrophys. J. 93 (1941) 70–83, http://dx.doi.org/10.1086/144246.
[68] P. Isola, J.-Y. Zhu, T. Zhou, A.A. Efros, Image-to-image translation with conditional adversarial networks, in: CVPR, 2017.
[69] J.Y. Zhu, T. Park, P. Isola, A.A. Efros, Unpaired image-to-image translation using cycle-consistent adversarial networks, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2223–2232.
[70] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[71] C. Chu, A. Zhmoginov, M. Sandler, Cyclegan, a master of steganography, 2017, arXiv preprint arXiv:1712.02950.
[72] A.X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, F. Yu, ShapeNet: An Information-Rich 3D Model Repository, Technical Report, Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015, arXiv:1512.03012 [cs.GR].
[73] R. Hess, Blender Foundations: The Essential Guide to Learning Blender 2.6, Focal Press, 2010.
[74] B. Shi, Z. Wu, Z. Mo, D. Duan, S.-K. Yeung, P. Tan, A benchmark dataset and evaluation for non-lambertian and uncalibrated photometric stereo, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3707–3716.
[75] Y. Ba, A. Gilbert, F. Wang, J. Yang, R. Chen, Y. Wang, L. Yan, B. Shi, A. Kadambi, Deep shape from polarization, in: European Conference on Computer Vision, Springer, 2020, pp. 554–571.
[76] M. Corsini, M. Dellepiane, F. Ponchio, R. Scopigno, Image-to-geometry registration: a mutual information method exploiting illumination-related geometric properties, in: Computer Graphics Forum, vol. 28, no. 7, Wiley Online Library, 2009, pp. 1755–1764.
[77] W. Jakob, S. Speierer, N. Roussel, D. Vicini, Dr.Jit: A just-in-time compiler for differentiable rendering, Trans. Graph. (Proc. SIGGRAPH) 41 (4) (2022) http://dx.doi.org/10.1145/3528223.3530099.
[78] C. Ye, L. Qiu, X. Gu, Q. Zuo, Y. Wu, Z. Dong, L. Bo, Y. Xiu, X. Han, Stablenormal: Reducing diffusion variance for stable and sharp normal, ACM Trans. Graph. 43 (6) (2024) 1–18.
[79] S. Woo, S. Debnath, R. Hu, X. Chen, Z. Liu, I.S. Kweon, S. Xie, Convnext v2: Co-designing and scaling convnets with masked autoencoders, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 16133–16142.
[80] Z. Liu, H. Hu, Y. Lin, Z. Yao, Z. Xie, Y. Wei, J. Ning, Y. Cao, Z. Zhang, L. Dong, et al., Swin transformer v2: Scaling up capacity and resolution, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 12009–12019.
[81] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., An image is worth 16x16 words: Transformers for image recognition at scale, 2020, arXiv preprint arXiv:2010.11929.
[82] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin transformer: Hierarchical vision transformer using shifted windows, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 10012–10022.