IPOL IMAGE PROCESSING ON LINE

Published in Image Processing On Line on 2025–03–12.
Submitted on 2024–07–09, accepted on 2025–01–17.
ISSN 2105–1232 © 2025 IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials,
software, datasets and online demo at
https://doi.org/10.5201/ipol.2025.562

Specularity in NeRFs: A Comparative Study of Ref-NeRF and NRFF

Albert Barreiro[1], Roger Marí[1], Rafael Redondo[1],
Gloria Haro[2], Carles Bosch[3], David Berga[1]

[1]Eurecat, Centre Tecnològic de Catalunya, Multimedia Technologies, Barcelona, Spain
({albert.barreiro, roger.mari, rafael.redondo, david.berga}@eurecat.org)
[2]Universitat Pompeu Fabra, Barcelona, Spain
(gloria.haro@upf.edu)
[3]Universitat de Vic – Universitat Central de Catalunya, Vic, Spain
(carles.bosch@uvic.cat)

Communicated by Pablo Musé *Demo edited by* Albert Barreiro

Abstract

Neural Radiance Fields (NeRF) have emerged as a leading technology for 3D digitization, especially for their high accuracy and intricate detailing. Despite their advancements, early NeRF models struggle to handle reflections on specular surfaces effectively. To address this, alternative approaches such as Ref-NeRF and NRFF were proposed to improve fidelity in representing this physical phenomenon. This study compares these two models, providing an analysis of their effectiveness and limitations in dealing with complex specularities. We demonstrate that both methods struggle with inter-reflections and tend to model anisotropic specularities by altering the predicted surface normals.

Source Code

The source code and documentation for these algorithms are available from the web page of this article[1]. Usage instructions are included in the `README.md` file of the archive. The original implementations of the methods are available here: Ref-NeRF[2] and NRFF[3].
This is an MLBriefs article. The source code has not been reviewed!

Keywords: 3D reconstruction; view synthesis; photorealistic rendering; volume rendering; Neural Radiance Fields

[1]https://doi.org/10.5201/ipol.2025.562
[2]https://github.com/kakaobrain/NeRF-Factory
[3]https://github.com/imkanghan/nrff

1 Introduction

Novel View Synthesis addresses the challenge of reconstructing unseen views of a scene from a set of sparse input images. This field has experienced significant progress with the development of techniques such as Neural Radiance Fields (NeRF) [9] and its subsequent version Mip-NeRF [1], which have transformed the ability to represent complex scenes with high degree of photorealism. These neural rendering methods have marked a turning point in the field of 3D scene understanding and representation.

A neural radiance field is a neural network that represents a 3D continuous function in space describing the appearance and geometry properties of a scene or object. NeRFs can be used to extract 3D geometric models and render novel views from arbitrary points of view.

Despite the spectacular advances in NeRFs [3], accurately representing complex specular reflections —as in metallic or wet surfaces— is a persistent challenge [10, 4]. In fact, the light transmittance model used in most NeRF approaches, including Mip-NeRF [1], ignores the physical properties of the reflection phenomena, often leading to poor results in reflective surfaces as shown in Figure 1. Without an explicit physical modeling, the network commonly represents specular reflections by inducing geometrical distortions or an artificial foggy appearance inside the objects.

(a) Ground truth (b) NeRF (c) Mip-NeRF (d) Ref-NeRF (e) NRFF

Figure 1: Foundational models NeRF and Mip-NeRF fail to render highly specular surfaces accurately, also unable to represent different specular roughness of the stripes around. Ref-NeRF and NRFF are able to render substantially more accurate details based on physical modeling of the reflection phenomena.

In response to these limitations, the recent variants Ref-NeRF [10] and Neural Radiance Feature Field (NRFF) [4] proposed modifications to model more accurately the physical phenomena of specularity.

This paper analyzes the Ref-NeRF and NRFF models on different datasets of simple and complex specular materials. A comparison of rendering quality and convergence speed, along with an in-depth discussion of the strengths and limitations of the methods is provided.

2 Related Work

This section first reviews the theoretical basis of NeRF, followed by an analysis of the Ref-NeRF and NRFF methods and how they differ from other approaches which do not explicitly model specular reflections in a physically based manner. Both Ref-NeRF and NRFF propose a reparameterization of the direction of reflection $\widehat{\boldsymbol{\omega}}_r$ based on the view direction $\widehat{\mathbf{d}}$ with respect to the predicted surface normal $\widehat{\mathbf{n}}$, i.e., $\widehat{\boldsymbol{\omega}}_r = 2(-\widehat{\mathbf{d}} \cdot \widehat{\mathbf{n}})\widehat{\mathbf{n}} + \widehat{\mathbf{d}}$, which is used to encode the radiance of the environment map in the direction $\widehat{\boldsymbol{\omega}}_r$ —encoded in a different way in each method. Note that this reparameterization ignores interreflections, but it assumes a single reflection rebound.

While other significant subsequent projects have been published very recently, e.g., NeRF-Casting [11] or UniSDF [12], they fall outside the scope of this work.

2.1 NeRF Preliminaries

Neural Radiance Fields or NeRF [9] estimate a parameterized continuous scene representation based on Multilayer Perceptrons (MLPs) and ray tracing principles. The network takes as input a 5D vector, consisting of a 3D location $\mathbf{x} = (x, y, z)$ and a 2D viewing direction $\mathbf{d} = (\alpha, \beta)$, and estimates a rendered color $\mathbf{c} = (r, g, b)$ and volume density τ for each camera ray. The input information is obtained from the estimated camera poses of the images. The volume density defines the geometry of the scene and can be interpreted as the probability of a 3D point belonging to an opaque non-empty object.

NeRF uses two networks, "coarse" and "fine", simultaneously optimized. This strategy allows for more efficient sampling of the scene, as the fine network can focus on regions of the scene where more detail is needed, while the coarse network provides a rough approximation of the entire scene. This procedure can be outlined as follows

$$\begin{aligned} (\tau, \mathbf{z}) &= F_{\theta_1}(\gamma_x(\mathbf{x})), \\ \mathbf{c} &= F_{\theta_2}(\gamma_d(\mathbf{d}), \mathbf{z}), \end{aligned} \tag{1}$$

where θ_1 and θ_2 are the MLP parameters, γ_x and γ_d the positional encoding functions for the position $\mathbf{x}$ and the direction $\mathbf{d}$, respectively, and $\mathbf{z}$ a hidden representation of the coarse model.

The estimated color per ray, denoted by $\widehat{C}(\mathbf{r})$, is obtained by a discretized integral quadrature along the camera ray $\mathbf{r}$, whose continuous representation can be denoted as $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$, where $\mathbf{o}$ is the camera center and $\mathbf{d}$ is the ray direction.

$$\begin{aligned} \widehat{C}(\mathbf{r}) &= \sum_{i=1}^{N} \mathrm{T}_i (1 - exp(-\tau_i \delta_i)) \mathbf{c}_i, \\ \text{where } \mathrm{T}_i &= exp(-\sum_{j=1}^{i-1} \tau_j \delta_j), \end{aligned} \tag{2}$$

and $\delta_i = t_{i+1} - t_i$ denotes the distance between adjacent points along the ray. $\mathbf{c}_i$ and τ_i represent the color and density of ray $\mathbf{r}$ at position t_i, respectively. The "transmittance" T_i defines the probability of the ray traveling from t_1 to t_{i-1} without intersecting other particles. The term $\mathrm{T}_i(1 - exp(-\tau_i \delta_i))$ serves as the rendering weight, determining the contribution of each sample to the final color.

NeRF employs a stratified sampling technique to achieve a smooth and continuous representation from a discrete set of samples. This approach involves dividing the space into evenly-spaced bins and randomly selecting a single sample within each bin. The stratified sampling ensures that multiple positions per bin in each ray are evaluated during the optimization, approximating an uninterrupted representation of the scene.

The MLP parameters are optimized by minimizing the sum of squared errors between a collection of images I_i and their corresponding rendered outputs. The cost function L is as follows

$$L = \sum_{ij} \left[\left\| \widehat{C}_c(\mathbf{r}_{ij}) - C(\mathbf{r}_{ij}) \right\|_2^2 \right] + \left[\left\| \widehat{C}_f(\mathbf{r}_{ij}) - C(\mathbf{r}_{ij}) \right\|_2^2 \right], \tag{3}$$

where $C(\mathbf{r}_{ij})$, $\widehat{C}_c(\mathbf{r}_{ij})$ and $\widehat{C}_f(\mathbf{r}_{ij})$ are the ground truth, the coarse RGB prediction, and the fine RGB prediction of ray j in image I_i.

2.2 Ref-NeRF Fundamentals

The Ref-NeRF method improves the representation of specular reflections by modeling the outgoing radiance model as

$$\mathbf{c} = \gamma(\mathbf{c}_d + \mathbf{s} \odot \mathbf{c}_s), \tag{4}$$

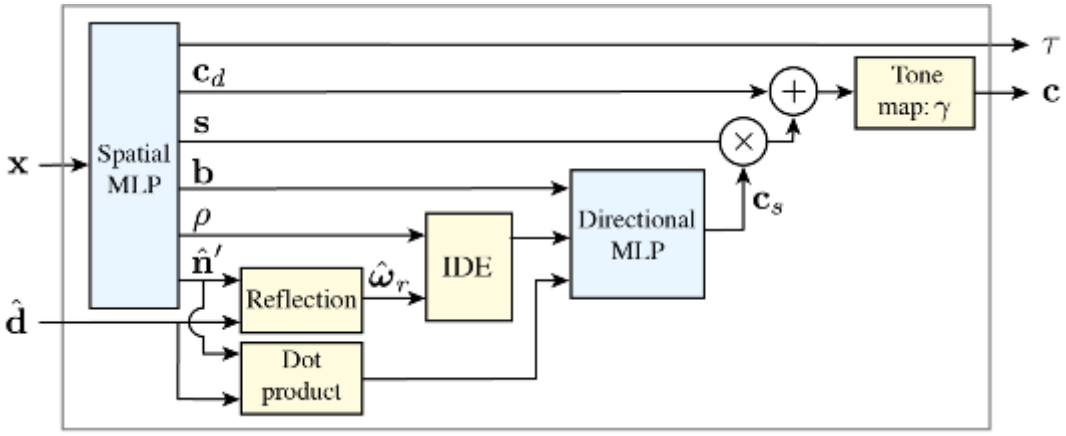

Figure 2: Block diagram of the Ref-NeRF architecture. The blue blocks correspond to two MLPs for predicting the spatial diffuse and directional specular color components. The yellow blocks stand for physically-constrained mathematical operations. Reproduced from [10].

where $\mathbf{c}_d$ is the diffuse color (intrinsic to objects), $\mathbf{c}_s$ is the specular color (caused by reflections) and $\mathbf{s}$ modulates the intensity and color of highlights. Element-wise multiplication is denoted by $\odot$ and γ is a predefined tone mapping function. Figure 2 shows the original block diagram of the Ref-NeRF architecture.

The Ref-NeRF blocks also predict auxiliary physical variables such as the surface normal $\widehat{\mathbf{n}}'$ and the material roughness ρ, which mainly govern reflection phenomena.

The spatial MLP takes as input the spatial location $\mathbf{x}$, which is subject to a Positional Encoding (PE) to improve high-frequency details, as in the original NeRF [9]. For each 3D point $\mathbf{x}$, it predicts the diffuse color $\mathbf{c}_d$, the tint color $\mathbf{s}$, a bottleneck feature vector $\mathbf{b}$ with dimensionality 128, the roughness scalar ρ, and the normal vector $\widehat{\mathbf{n}}'$.

The directional MLP predicts the specular color component $\mathbf{c}_s$. To this end, it mainly uses the bottleneck vector $\mathbf{b}$ and an Integrated Directional Encoding (IDE) which encodes the distribution of reflection directions towards the environment. For that, it utilizes a closed-form of spherical harmonics convolved with the von Mises-Fisher (vMF) distribution [6]. It is analytically obtained from the direction of reflection $\widehat{\boldsymbol{\omega}}_r$ and the material roughness ρ. The directional MLP additionally takes the dot product of $\widehat{\mathbf{n}}'$ and the viewing direction $\widehat{\mathbf{d}}$ as input to account for Fresnel effects [5], so that the angle between the $\widehat{\mathbf{n}}'$ and $\widehat{\mathbf{d}}$ affects the strength of the reflection.

Ref-NeRF also introduces two auxiliary loss terms to regularize the surface normals, which are crucial to model the outgoing radiance and reflection. Physically consistent normals are expected to be perpendicular to the local surface plane and point outwards. The Ref-NeRF auxiliary loss terms are formulated as:

$$\mathcal{L}_{\mathrm{p}} = \sum_i w_i \left\| \widehat{\mathbf{n}}_i - \widehat{\mathbf{n}}'_i \right\|^2, \quad \mathcal{L}_{\mathrm{o}} = \sum_i w_i \max\left(0, \widehat{\mathbf{n}}'_i \cdot \widehat{\mathbf{d}}\right)^2. \tag{5}$$

On the one hand, $\mathcal{L}_{\mathrm{p}}$ penalizes the differences between the predicted normals $\widehat{\mathbf{n}}'$ and those derived from the gradient of the predicted density, i.e., $\widehat{\mathbf{n}}$. On the other hand, $\mathcal{L}_{\mathrm{o}}$ basically encourages normals pointing outward. Finally, w_i are the rendering weights [10] that model the visible surface, based on the estimated transmittance and density.

2.3 NRFF Fundamentals

The Neural Radiance Feature Field method (NRFF) has shown proficiency in encoding view-dependent effects, enabling superior rendering quality on diverse datasets [9, 8, 7]. The NRFF architecture, il-

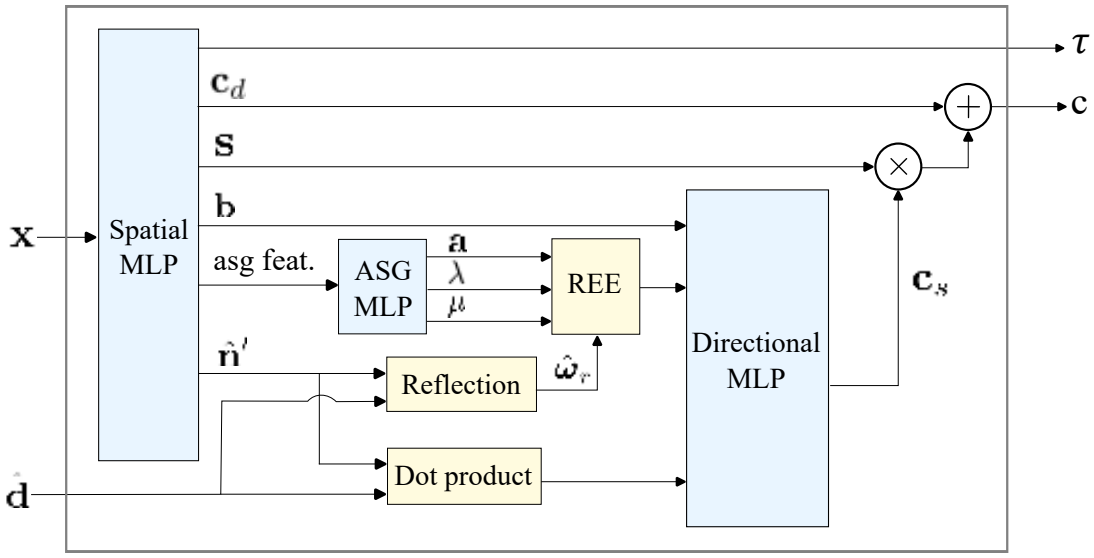

Figure 3: Block diagram of the NRFF architecture. It introduces a neural network to learn features of the Anisotropic Spherical Gaussian (ASG) as well as a custom Rendering Equation Encoding (REE) operating in the latent space.

lustrated in Figure 3, uses the same radiance model as Ref-NeRF, Equation (4), to model different color components, however it proposes architectural improvements in different ways.

The spatial MLP uses a multiscale tensor decomposition (MTD) instead of PE to represent the scene in a coarse-to-fine scheme. MTD is proposed for efficiency in reconstructing scene geometry and appearance. The key difference with respect to the Ref-NeRF counterpart is that the NRFF spatial MLP predicts features of Anisotropic Spherical Gaussians (ASG) to represent the reflection at each point. In particular, it predicts $N = 8 \times 16$ lobes of ASGs on a unit sphere for each point, resulting in a vector of 128 values. The ASG features are then input to an auxiliary ASG MLP, that converts them into an anisotropic reflections bottleneck vector $\mathbf{a}$ of length 2, as well as 2 scalar coefficients, λ and μ, modeling the shape of the ASG in orthogonal directions.

At each point $\mathbf{x}$, NRFF performs the Rendering Equation Encoding (REE) using the ASG MLP outputs and the direction of reflection $\widehat{\boldsymbol{\omega}}_r$. The REE integrates all the specular contributions in $\mathbf{x}$ in the latent space

$$REE(\widehat{\boldsymbol{\omega}}_r, \boldsymbol{a}, \lambda, \mu) = \sum_{i=0}^{N-1} \mathbf{a}_i S(\widehat{\boldsymbol{\omega}}_r; \boldsymbol{\omega}_i) \exp\left(-\lambda_i (\widehat{\boldsymbol{\omega}}_r \cdot \boldsymbol{\omega}_i^{\lambda})^2 - \mu_i (\widehat{\boldsymbol{\omega}}_r \cdot \boldsymbol{\omega}_i^{\mu})^2\right), \tag{6}$$

where $\mathbf{a}_i$ is the bottleneck of the i-th ASG, $S(\widehat{\boldsymbol{\omega}}_r; \omega_i) = \max(\widehat{\boldsymbol{\omega}}_r \cdot \boldsymbol{\omega}_i, 0)$ is a smooth term, and $\lambda_i, \mu_i > 0$ are the bandwidth coefficients (roughness) related to the orthogonal axes $\boldsymbol{\omega}_i^{\lambda}, \boldsymbol{\omega}_i^{\mu}$ shaping and orienting the ASG. The set of $\widehat{\boldsymbol{\omega}}_i$ covers 360 degrees (the unit sphere) and each direction $\widehat{\boldsymbol{\omega}}_i$ is associated with an ASG.

Ultimately, the NRFF directional MLP ingests an $8 \times 16 \times 2$ REE vector replacing the roughness and IDE in Ref-NeRF.

In the loss function, NRFF keeps the auxiliary term $\mathcal{L}_{\text{o}}$ from Ref-NeRF, Equation (5), and introduces a new regularization term $\mathcal{L}_{\text{f}}$ as follows

$$\mathcal{L}_{\text{f}} = \beta \frac{1}{M} \sum_{i=0}^{M-1} |\mathbf{F}_{\tau}^{i}|, \tag{7}$$

where M is the number of features in the density field representation, and $\mathbf{F}_{\tau}$ is the density field feature map. This results in a sparse feature representation. Sparse representations enhance model

interpretability by highlighting the most important features and can mitigate overfitting by removing less relevant or redundant information from the feature space.

3 Methodology

This section describes the datasets, code, and hardware used in our comparative study.

3.1 Datasets

Two objects from the *Shiny Blender* dataset [10] were selected and one more was designed for this evaluation. It is important to note that, due to hardware constraints, the different methods were trained at half the original image resolution. Example views from the different datasets are shown in Figure 4.

Ball. It consists of 300 images —100 for training, 200 for testing— with an original resolution of 800×800 pixels. It consists of a single reflective ball with bands of different roughness that affect the sharpness of the reflection. Only two different values of roughness are used.

Toaster. It consists of 300 images —100 for training, 200 for testing— with an original resolution of 800×800 pixels. The toaster has a more complex structure. It includes specular, non-specular materials, and inter-reflections, a physical phenomenon which escapes from the single-ray modelling assumed by the herein studied models.

Anisotropic ball. It consists of 100 images —67 for training and 33 for testing— with an original resolution of 1080×1080 pixels. Created with Blender[4], it simulates a very high level of complexity with textured anisotropic reflections, a property unexplored in Ref-NeRF and NRFF. Additionally, the object has a strip of higher roughness. It is publicly available here[5].

(a) Ball (b) Toaster (c) Anisotropic ball

Figure 4: Views of the 3 objects studied in this work.

3.2 Code

Our experiments used the Ref-NeRF implementation from NeRF-Factory[6], in PyTorch Lightning, instead of the original implementation in JAX[7]. This implementation follows the hyperparameter

[4] https://www.blender.org/
[5] https://zenodo.org/records/12568381
[6] https://github.com/kakaobrain/NeRF-Factory
[7] https://github.com/google-research/multinerf

settings specified in the original paper. However, due to a decreased batch size of 1800, to load the model in a single GPU, the number of epochs was increased up to 50, the learning rate reduced to 5×10^{-6}, and the warm-up period set to 2500.

In the case of NRFF, the original code (NRFF[8]) provided by the authors was used. The batch size was set to 2000 along with 150 epochs, while the rest of the configuration remained the same. It's worth noting that while NRFF runs faster per epoch, it requires a greater number of epochs to achieve convergence compared to other methods. Furthermore, for the *Ball* object, the hyperparameter of loss $\mathcal{L}_o$ in Equation (5) was increased to 0.9 to achieve optimal convergence.

Consequently, some results may differ from those reported in the original papers. Additionally, metrics and visualization functionalities were implemented in both codes to facilitate comparison.

3.3 Demo

An online demo[9] has been developed to complement this study. This demo offers a comprehensive comparison between Ref-NeRF and NRFF. Users can select from the three distinct objects presented in Section 3.1. The demo provides side-by-side comparisons of renderings, surface normals, and quantitative metrics including PSNR, SSIM [13] and LPIPS [14]. Additionally, the rendering view allows for comparison with ground truth images.

This demo serves to enhance research transparency and reproducibility by enabling readers to independently evaluate and compare the output of the models without the need for time-consuming training processes or specialized hardware.

3.4 Hardware

The models were trained on an Intel Core i9-10900X CPU @ 3.70GHz equipped with an NVIDIA GeForce RTX 3090 graphics card with 24.5 GB memory. The Ref-NeRF implementation used Pytorch Lightning 1.8.0 and CUDA 12.3. The GPU memory usage for a batch size of 1800 rays per iteration was around 22.5 GB. The NRFF implementation used PyTorch 2.2.0 and CUDA 12.3. A batch size of 2000 rays required 21 GB.

4 Experiments

This section presents both qualitative and quantitative results comparing the performance of Ref-NeRF and NRFF on the three object scenes introduced in Section 3.1.

Quantitative results are assessed by means of PSNR, SSIM and LPIPS, as shown in Table 1. Both Ref-NeRF and NRFF demonstrate exceptional performance on the *Ball* object, with Ref-NeRF being slightly better. For the *Toaster* object, NRFF shows better overall quality. Interestingly, both methods show a similar and remarkable performance in the *Anisotropic Ball*, in spite of not being specifically designed for such complex patterns of reflectance.

Qualitative analysis in Figure 5 shows high-quality RGB renderings for both models. Normals are also highly accurate, although NRFF exhibits some issues at the edges where numerous floating points are present. This is likely due to their filtering approach. The model computes an alpha value for each point based on its density $\alpha_i = 1 - \exp(-\sigma_i \delta_i)$, discarding points with values below 10^{-3}.

Additionally, the rendering weights shown in Figure 5 reveal that Ref-NeRF exhibits minor perturbations in the roughness band. It is important to note that the discrepancies observed in our

[8] https://github.com/imkanghan/nrff
[9] https://ipolcore.ipol.im/demo/clientApp/demo.html?id=562

	Ball			Toaster			Anisotropic ball		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Ref-NeRF	50.76	0.999	0.0036	25.85	0.925	0.0789	41.75	0.995	0.0143
NRFF	50.0	0.998	0.0055	28.52	0.951	0.0651	40.06	0.994	0.0223

Table 1: Comparison of Ref-NeRF and NRFF in two open-source objects, i.e. Ball and Toaster, and an Anisotropic ball created by design for this work across three different objective quality metrics.

results compared to those presented in the Ref-NeRF paper are due to the modifications discussed in Section3.2.

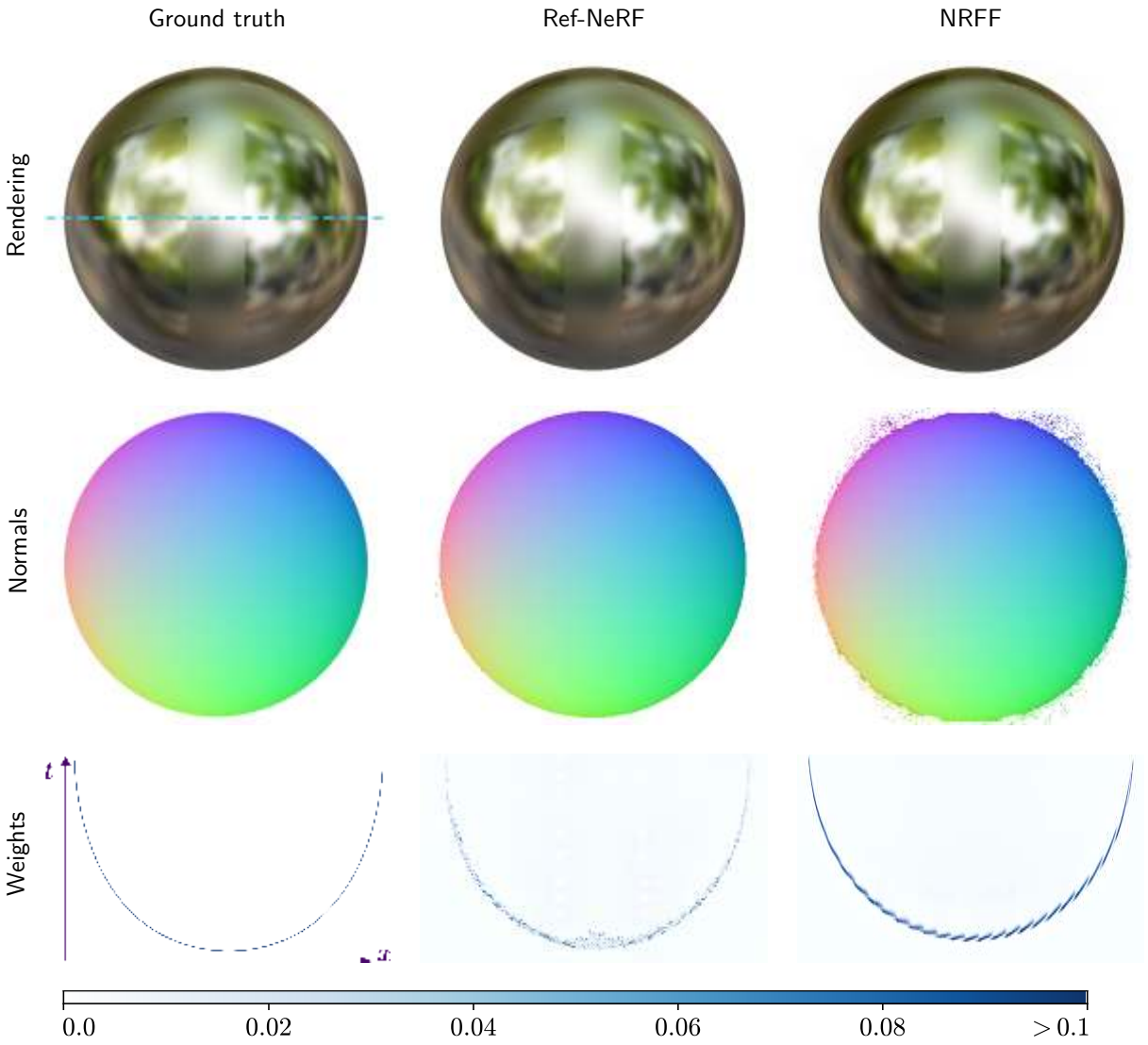

Figure 5: Ref-NeRF and NRFF comparison on the *Ball* object. Top to bottom: RGB rendering, predicted normals and rendering weights. The bottom row plots a cross-section of the rendering weights predicted by each model, sampled along the blue-dashed line.

Figure 6 highlights the difficulties of Ref-NeRF and NRFF to handle inter-reflections such as those in the *Toaster* object. Despite the correct RGB renderings, the underlying depth maps reveal an erroneous geometry at the front of the toaster that matches the inter-reflection crossed by the dashed blue line. The rendering weights show that NRFF bends the surface inward while Ref-NeRF produces a foggy area to explain the inter-reflection. This limitation arises from the fact that these models are not based on ray tracing and consider only a single bounce of light. As a result, inter-reflections lead to inaccuracies in geometry reconstruction.

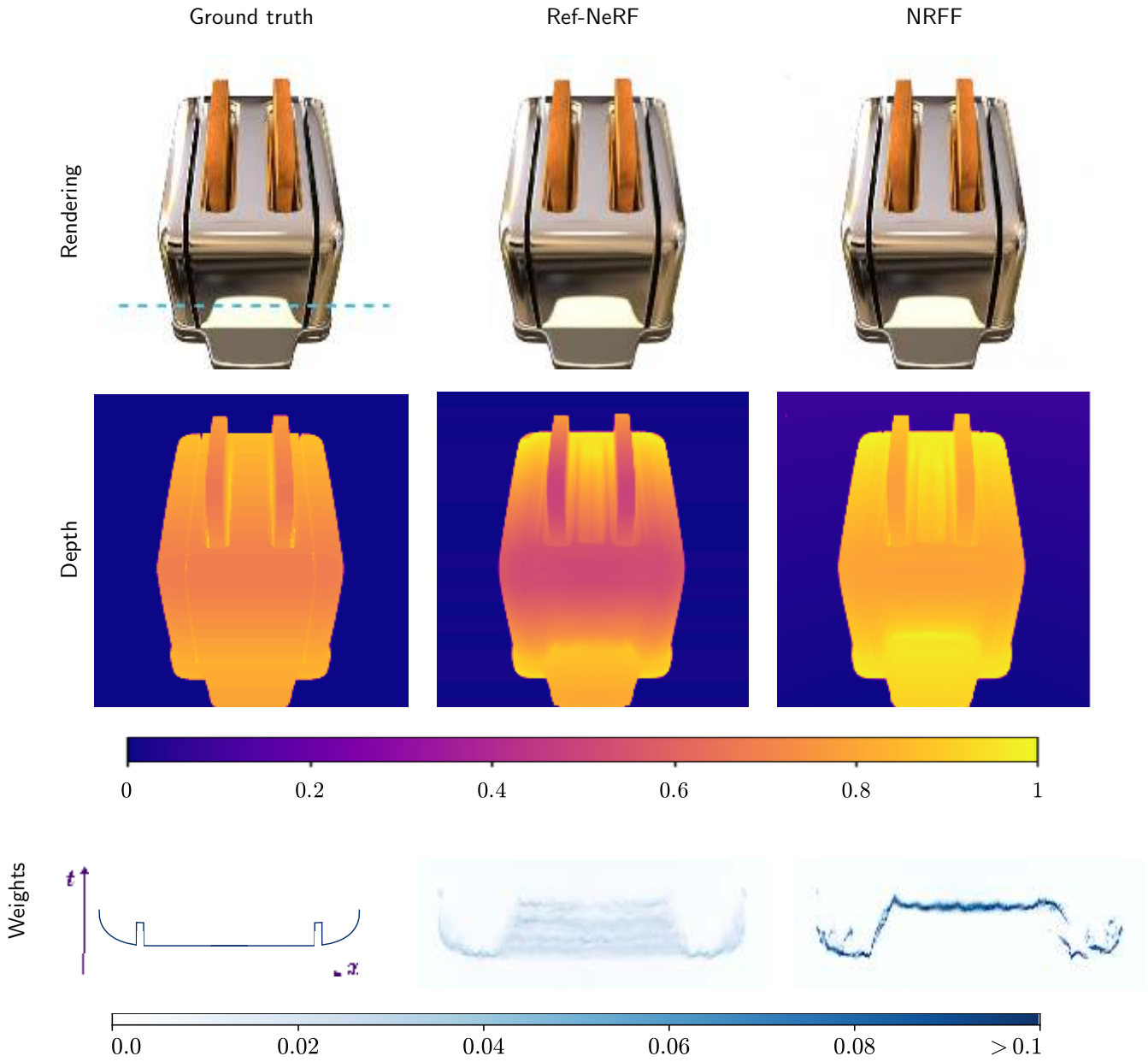

Figure 6: Ref-NeRF and NRFF comparison on the *Toaster* object. Top to bottom: RGB rendering, depth maps, and rendering weights. The bottom row plots a cross-section of the rendering weights predicted by each model, sampled along the blue-dashed line. Both methods fail to represent inter-reflections.

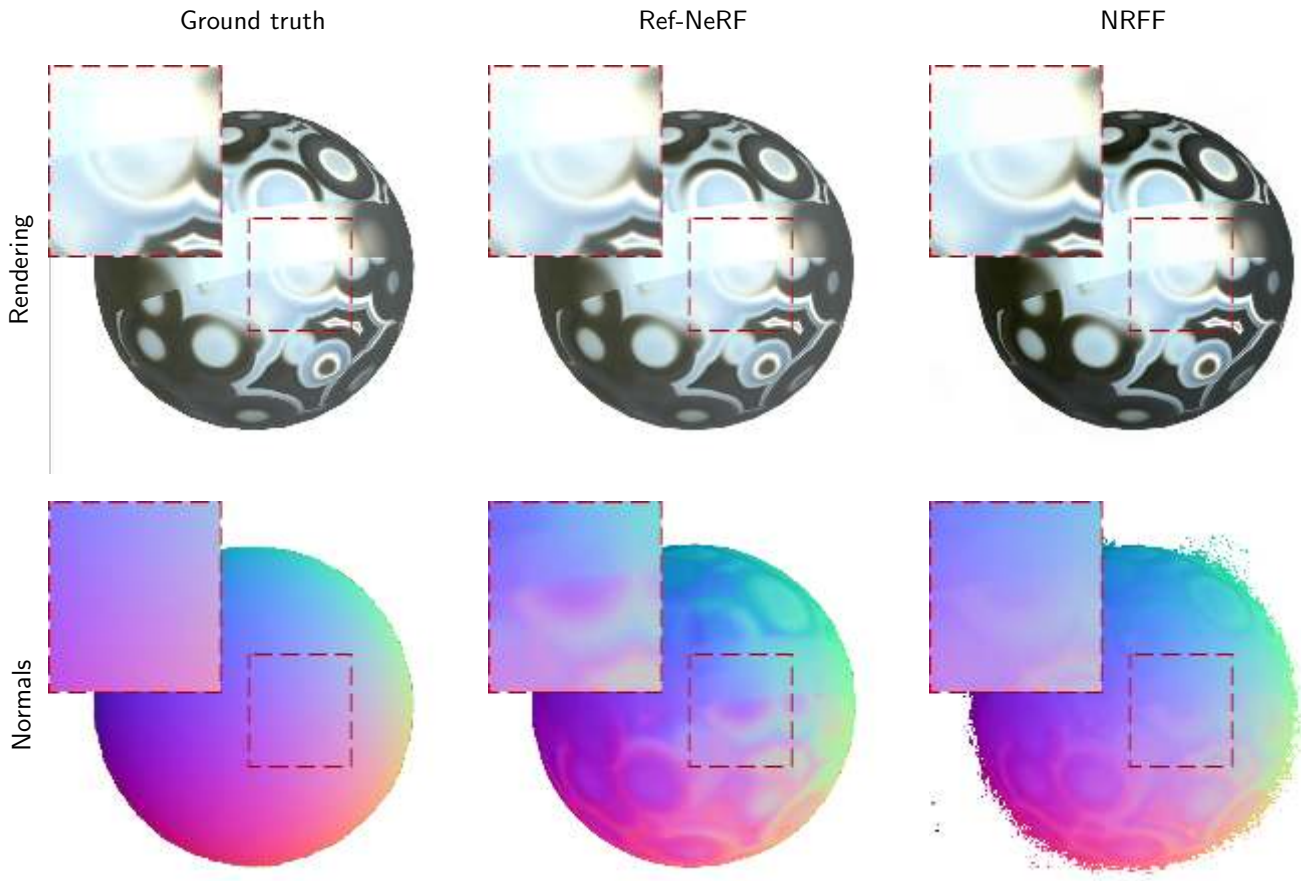

Figure 7: Ref-NeRF and NRFF comparison on the *Anisotropic Ball* object. Top to bottom: RGB rendering and surface normals. Both methods wrongly represent anisotropic reflections by altering geometry.

Figure 7 evidences the difficulty of both methods to handle anisotropic reflections in the *Anisotropic Ball* object. In this case the geometry of the ball is also artificially altered to achieve an accurate RGB rendering. This behavior is most noticeable in regions with high anisotropy. The discrepancy between the underlying geometry and the final rendered output suggests that the methods differ in their ability to compensate for geometric alterations in the rendering process. In any case, none of the methods is able to correctly reconstruct anisotropic surfaces.

Further examination of how both Ref-NeRF and NRFF internally represent or learn each of the color components, as formulated in (4), is of great relevance to understand the response of the inner architecture blocks. As so, the diffuse component in Figure 8 should represent Lambertian properties of the material, independent of lighting conditions or viewing angle. In the case of specular metallic surfaces the diffuse component is normally nonexistent (black). Ref-NeRF can accurately simulate this property, while NRFF fails substantially.

The tint color in Figure 8 modulates the color and intensity of highlights. The tint in the *Ball* and the *Toaster* should be mostly white. The specular color should represent the reflected surrounding environment and illumination. The final RGB color is the composite result, combining the diffuse color with the tinted specular reflections.

Figure 8 shows that NRFF exhibits significant limitations in accurately representing diffuse, tint, and specular components in both objects. Some of these inaccuracies, however, are compensated when the components are finally combined. This dysfunctionality is due to the lack of per-component sigmoid activation functions that would constrain the dynamic range of the specular and tint components between 0-1. In practice this leads the NRFF to work with negative radiance values, which

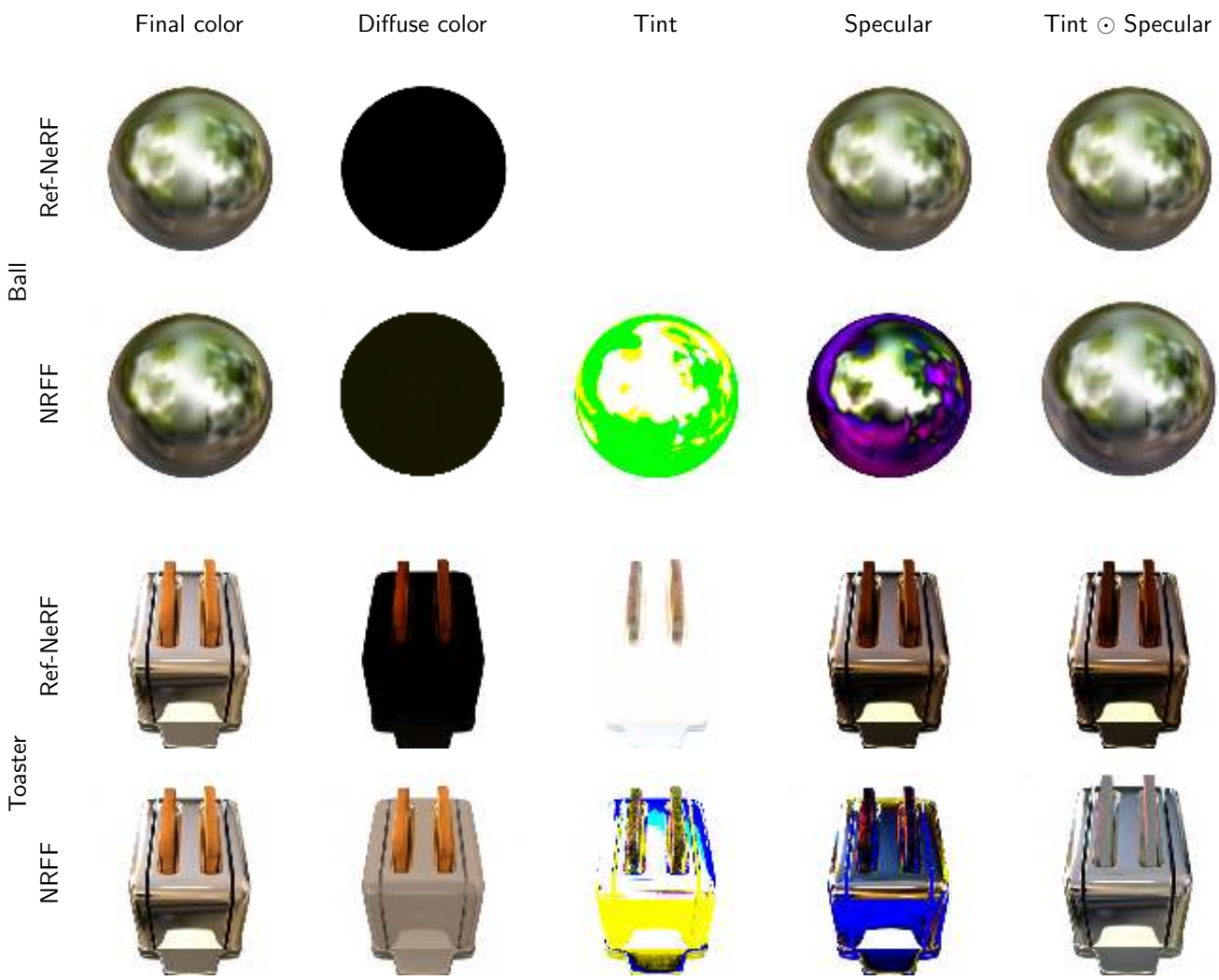

Figure 8: Visualization of individual color components from (4). From left to right: final render ($\mathbf{c}$), diffuse component ($\mathbf{c}_d$), tint ($\mathbf{s}$), specular component ($\mathbf{c}_s$), and tint $\odot$ specular ($\mathbf{s} \odot \mathbf{c}_s$). Note that NRFF predicts negative numbers for tint and specular without a clear physical interpretation, and so they have been simply normalized for visualization purpose.

are difficult to interpret without a strict physical meaning.

Ref-NeRF, while superior overall, still presents inaccuracies. Notably, in the *Toaster* object, it erroneously assigns a certain amount of specularity to the bread slices, indicating incomplete comprehension of the object's geometric properties.

As NRFF demonstrated faster convergence compared to its predecessor, i.e. TensorRF [2], it is interesting to analyze the optimization time of both, NRFF and Ref-NeRF. Table 2 shows that NRFF achieves faster convergence in the three objects, partly due to NRFF excluding points with low weight from being processed through the network.

	Ball	Toaster	Anisotropic ball
Ref-NeRF	2.44	2.74	2.86
NRFF	2.31	2.13	1.78

Table 2: Comparison of model execution times across the considered image collections (in days).

5 Conclusion

We have demonstrated that Ref-NeRF shows superior quantitative results and better handles geometric inaccuracies during rendering. However, both models struggle with inter-reflections and anisotropic surfaces as they consider only a single light bounce and isotropic materials.

While NRFF converges faster per step, it requires more steps overall and produces less accurate specular reflections.

Our study highlighted the trade-offs between rendering quality and convergence speed and, more importantly, the limitations of the models. The results show that improvements are still needed in the physical modeling of reflections in NeRF approaches, so that the geometry of objects is not distorted to account for specularities in the RGB renderings.

Acknowledgments

This work was financially supported by the Catalan Government through the funding grant ACCIÓ-Eurecat (Project TRAÇA: "AI4Heritage" 2023-2024). Additionally, Albert Barreiro Díaz is a fellow of Eurecat's "Vicente López" PhD grant program.

References

[1] 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 IEEE/CVF International Conference on Computer Vision, 2021, p. 5855–5864, https://doi.org/10.1109/ICCV48922.2021.00580.

[2] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su, *TensoRF: Tensorial Radiance Fields*, in European Conference on Computer Vision, Springer, 2022, pp. 333–350, https://doi.org/10.1007/978-3-031-19824-3_20.

[3] K. Gao, Y. Gao, H. He, D. Lu, L. Xu, and J. Li, *NeRF: Neural Radiance Field in 3D Vision, a Comprehensive Review*, ArXiv Preprint ArXiv:2210.00379, (2022), https://doi.org/10.48550/arXiv.2210.00379.

[4] K. Han and W. Xiang, *Multiscale Tensor Decomposition and Rendering Equation Encoding for View Synthesis*, in IEEE / CVF Computer Vision and Pattern Recognition Conference, 2023, pp. 4232–4241, https://doi.org/10.1109/CVPR52729.2023.00412.

[5] J. Kautz and M. D. McCool, *Approximation of Glossy Reflection with Prefiltered Environment Maps*, in Graphics Interface, 2000, pp. 119–126.

[6] T. Kitagawa and J. Rowley, *Von Mises-Fisher Distributions and their Statistical Divergence*, ArXiv Preprint ArXiv:2202.05192, (2022), https://doi.org/10.48550/arXiv.2202.05192.

[7] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun, *Tanks and Temples: Benchmarking Large-Scale Scene Reconstruction*, ACM Transactions on Graphics (ToG), 36 (2017), pp. 1–13, https://doi.org/10.1145/3072959.3073599.

[8] L. Liu, J. Gu, K. Zaw Lin, T.-S. Chua, and C. Theobalt, *Neural Sparse Voxel Fields*, Advances in Neural Information Processing Systems, 33 (2020), pp. 15651–15663.

[9] 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*, in European Conference on Computer Vision, 2020, https://doi.org/10.1007/978-3-030-58452-8_24.

[10] D. Verbin, P. Hedman, B. Mildenhall, T. Zickler, J. T. Barron, and P. P. Srinivasan, *Ref-NeRF: Structured View-Dependent Appearance for Neural Radiance Fields*, in IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, p. 5491–5500, https://doi.org/10.1109/CVPR52688.2022.00541.

[11] D. Verbin, P. P. Srinivasan, P. Hedman, B. Mildenhall, B. Attal, R. Szeliski, and J. T. Barron, *NeRF-Casting: Improved View-Dependent Appearance with Consistent Reflections*, 2024, https://doi.org/10.1145/3680528.3687585.

[12] F. Wang, M.-J. Rakotosaona, M. Niemeyer, R. Szeliski, M. Pollefeys, and F. Tombari, *UniSDF: Unifying Neural Representations for High-Fidelity 3D Reconstruction of Complex Scenes with Reflections*, ArXiv Preprint ArXiv:2312.13285, (2023), https://doi.org/10.48550/arXiv.2312.13285.

[13] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, *Image Quality Assessment: from Error Visibility to Structural Similarity*, IEEE Transactions on Image Processing, 13 (2004), pp. 600–612, https://doi.org/10.1109/TIP.2003.819861.

[14] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, *The Unreasonable Effectiveness of Deep Features as a Perceptual Metric*, in IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 586–595, https://doi.org/10.1109/CVPR.2018.00068.