Volume 40 (2021), Number 4 pp. 1–13 COMPUTER GRAPHICS *forum*

NeRF-Tex: Neural Reflectance Field Textures

H. Baatz[1,2] J. Granskog[2] M. Papas[3] F. Rousselle[2] J. Novák[2]

baatzh@student.ethz.ch jgranskog@nvidia.com mpapas@ethz.ch frousselle@nvidia.com jnovak@nvidia.com

[1]ETH Zurich [2]NVIDIA

Abstract

We investigate the use of neural fields for modeling diverse mesoscale structures, such as fur, fabric, and grass. Instead of using classical graphics primitives to model the structure, we propose to employ a versatile volumetric primitive represented by a neural reflectance *field (NeRF-Tex), which jointly models the geometry of the material and its response to lighting. The NeRF-Tex primitive can be instantiated over a base mesh to "texture" it with the desired meso and microscale appearance. We condition the reflectance field on user-defined parameters that control the appearance. A single NeRF texture thus captures an entire space of reflectance fields rather than one specific structure. This increases the gamut of appearances that can be modeled and provides a solution for combating repetitive texturing artifacts. We also demonstrate that NeRF textures naturally facilitate continuous level-of-detail rendering. Our approach unites the versatility and modeling power of neural networks with the artistic control needed for precise modeling of virtual scenes. While all our training data is currently synthetic, our work provides a recipe that can be further extended to extract complex, hard-to-model appearances from real images.*

CCS Concepts

• ***Computing methodologies*** → ***Reflectance modeling; Volumetric models; Neural networks;*** *Ray tracing;*

1. Introduction

Recent progress in neural rendering has demonstrated great potential as illustrated by the unprecedented realism of portrait images rendered with generative adversarial networks [KLA*20]. This realism stems in part from an accurate reproduction of perceptually important mesoscale details such as wrinkles, freckles, and facial hair. Unfortunately, these neural techniques are of limited practical value for production rendering as most permit only limited artistic control, and rarely generalize beyond the initial training context.

In contrast, traditional rendering pipelines offer fine-grained artistic control and are built on the concept of asset reuse. This flexibility has led to a rich ecosystem of authoring tools and rendering algorithms targeting a wide range of applications. Nevertheless, fuzzy mesoscale details (e.g. grains or fur) have proven to be challenging for traditional graphics approaches: polygons tend to be wasteful, curves have limited applicability and are hard to filter, voxels suffer from scalability issues, and their hierarchies are hard to edit.

The complementary nature of the respective strengths and weaknesses of neural and traditional approaches has led to a number of efforts to combine them. In this context, *neural radiance fields* (NeRF) [MST*20]—a technique that renders a neural scene representation using classical ray marching—offers a compelling trade-off: the learned neural representation efficiently captures details that are tedious to author, while the use of (local) ray marching enables seamless integration into ray tracing algorithms.

One limitation of NeRF is that it tends to struggle with delivering high-quality visuals and sufficient artistic control when modeling the entire scene. Some progress has been made to increase the quality (and rendering speed) by factoring information out of the MLP into the scene [BXS*20, GFWF20, LGL*20, HSM*21, SDZ*21] or by modeling the scene using multiple fields [OMT*21, LSS*21]. We draw inspiration from these works and propose to texture objects using a collection of fields, each of which is an instance of a parametric NeRF primitive—we denote these *NeRF textures*. We constrain the neural modeling to local mesoscale (and microscale) appearance only, as in the case of classical volumetric textures. The macroscale appearance, i.e. the overall shape, is defined using a base mesh with surface attributes that drive the appearance of the parametric NeRF texture. This retains the efficiency of neural representations while preserving full artistic control over the macro appearance and moderate control over the meso and micro appearance—as much as the trained parametric texture provides; see Figure 1 for illustration.

One of the key benefits of neural scene representations [TZN19, KMX*21, BMT*21] is that they facilitate filtered queries of the content. We harness this ability to permit continuous and consistent level-of-detail rendering of NeRF textures. While the rendering efficiency of our approach is not yet comparable to classical primitives, like polygons and curves, the inherent ability to perform filtered queries provides a distinct advantage, especially in scenes with non-trivial depth complexity.

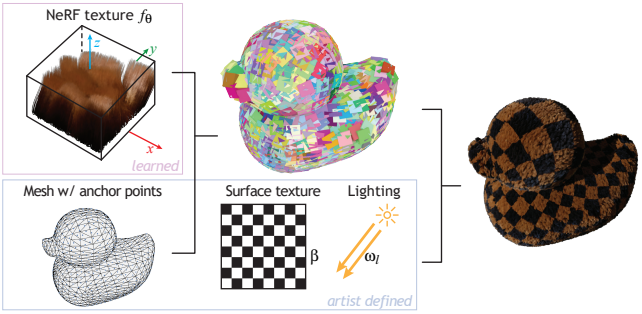

Figure 1: *We create a scene with mesoscale appearance using a learned NeRF texture, which is instantiated over a base mesh according to an artist-defined distribution of anchor points. The local appearance of the parametric NeRF texture is controlled using classical surface textures and lighting parameters.*

Another key benefit of NeRF textures is that they are capable of continuously representing an entire appearance space. We show results where the appearance of the parametric NeRF texture is driven by standard surface textures that modulate material albedo, length of fibers, or age of grass. NeRF textures are therefore more versatile than classical volume representations such as voxel grids.

While we employ only synthetic datasets in our results, the method is trained using 2D images and, therefore, conceptually extends to training from photographs. We conclude the article by discussing directions for improving the quality and outline developments that would facilitate deploying NeRF textures in production.

2. Related work

Next, we review prior works broadly categorized into classical approaches for modeling mesoscale appearance and neural methods that inspired our approach.

2.1. Mesoscale appearance

Much research has been focused on mesoscale models that capture (and control) the intricate meso structure and large-scale appearance of complex materials such as granular media, fabrics, fur, and skin. Fiber-based materials, such as cloth, can be captured from high-resolution CT scans and stored in memory-heavy volumetric representations [ZJMB11]. Zhao et al. [ZHRB13] propose a precomputation optimization that exploits the repetitive structure across patches and efficiently simulates multiple scattering. However, accurately modeling virtual cloth is an active area of research and many other approaches exist [SKZ11, ZJMB12, ZLB16, ACG*17, DXT17, MGZJ20]. Hair and fur generally rely on geometric curves to model the individual fibers. Photorealistic hair rendering requires simulating the intra-fiber transport, which is modeled with microfacet models [MJC*03], and inter-fiber transport, which can be explicitly path traced or approximated [ZYWK08, YTJR15, CBTB16, YSJR17].

Mesostructures in human faces are particularly challenging to model and render accurately. Pore-level details and wrinkles can be captured and stored in normal and displacement maps for rendering [SKU08, GTB*12]. Rendering layered translucent skin requires accurate simulation of sub-surface scattering, which is typically modeled using diffusion theory [JMLH01, DWd*08, DI11, HCJ13] or volumetric path tracing.

Various volumetric primitives have been devised for modeling mesostructures such as leaves and fur [KK89, Ney98, DN09]. See Koniaris et al. [KCYM14] for a survey of volumetric mesostructure texturing and Barnes et al. [BZ17] for an overview of patch-based synthesis research, which is closely tied to instantiation of volumetric primitives, especially for reducing repetition artifacts. Granular media, such as snow and sand, are also commonly modeled by instantiating volumetric primitives containing grain representations [MPH*15, MPG*16] with precomputed or approximated intra-grain light transport.

Appearance of certain objects can also be modeled by light fields [AB91, WAA*00] that have parts of the illumination baked in. Light fields can be captured from measurements [WJV*05] and rendered by slicing and interpolating high dimensional measurements [LH96]. For example, volumetric billboards [DN09] represent radiance fields with 3D textures. Despite their generality, it is not easy to author and control the appearance of light fields.

From these examples, one can already see that mesoscale appearance is challenging to model explicitly and requires various kinds of geometric primitives and textures, such as curves, voxel grids, and displacement maps. We seek a versatile primitive that could handle diverse structures well without excessive tailoring.

2.2. Neural rendering

Neural networks can learn to model complex materials from data, such as bidirectional texture functions (BTFs) [RJGW19, RGJW20, KMX*21] and BRDFs [HGC*20, SRRW21], hair [TCC*20, CRT20] and subsurface scattering [VKJ19, LHW21]. Neural textures learned from images can store view-dependent information for deferred rendering to allow novel view synthesis and scene editing [TZN19].

Neural networks have been used for representing light fields. Ren et al. [RWG*13] optimize multi-layer perceptrons (MLP) to enable faster rendering of global illumination in virtual scenes. Oechsle et al. [OMN*19] optimize a texture field that can be mapped to a shape using a radiance-predicting MLP, and Henzler et al. [HMR20] extend the idea to learn a stochastic 3D texturing space. Kallweit et al. [KMM*17] train radiance-predicting MLPs to efficiently render atmospheric clouds. MLPs have also been utilized to decrease the variance of Monte Carlo estimators [MMR*19, MRKN20].

Combining volumetric rendering, more specifically ray marching, with neural methods has recently gained immense popularity. Some learn geometric representations with signed distance fields [SZW19, TLY*21], others render color images from learned voxel representations [LSS*19]. Neural radiance fields (NeRF) [MST*20] learn a volumetric representation of real-world scenes from images that can be rendered from novel viewpoints. Several works improve the image quality and rendering speed of NeRF [LMW21, ZRSK20, LGL*20, HSM*21] whereas others focus on dynamic versions [PCPMMN20, PSB*20] or generative models [SLNG20, CMK*20]. Similar to our approach, Lombardi et al. [LSS*21] utilize a set of neural primitives to attain detail.

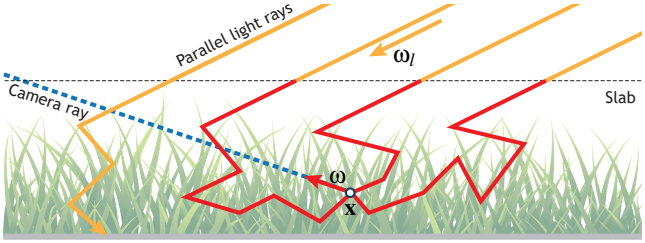

Figure 2: *The MLP is optimized to capture the volume density (green grass blades) and the fraction of parallel illumination, which enters a slab of the mesostructure, propagates to point* $\mathbf{x}$*, and continues in direction* ω *upon an interaction thereof. The fraction is modeled by the reflectance function g and illustrated by red paths.*

To reduce aliasing Barron et al. [BMT*21] integrate filtering into their neural field model by deriving a custom positional encoding. Our approach is simpler in that we input merely a filtering-kernel radius into the MLP and train it to perform filtered queries by blurring the reference images. Hierarchical neural representations offer another alternative for modeling multiple levels of detail [TLY*21, KMX*21].

Many recent works focus on capturing and predicting reflectance fields to enable relighting, and better integration into traditional rendering pipelines [BRTO*21, BXS*20, GFWF20, BBJ*20, SDZ*21]. We also learn neural reflectance fields for our texturing primitives. However, we instantiate the trained patches to apply mesoscale appearances to 3D meshes. Additionally, we learn a parametric model for each material class, instead of optimizing for static appearance, which allows us to synthesize multiple different appearances, including filtered appearance, with a single trained model.

3. Neural Reflectance Field Textures

Our work builds upon the concept of neural radiance fields, or NeRF [MST*20], deviating in three points from the original design.

1. We opt to model a *reflectance* field [SDZ*21] instead of a radiance field (Figure 2 illustrates the modeled transport), i.e. lighting is not baked in the neural representation but rather used as a conditional input.
2. Instead of using a single neural field to represent the entire scene, we use an assembly of neural fields to represent a layer of mesoscale structure on top of a base triangle mesh. Our approach is conceptually similar to volumetric textures [KK89, Ney98], with the distinction that we use a neural network to represent the content of the texture.
3. Our neural fields are parametric, i.e. they allow varying the density and reflectance fields as a function of artist-friendly parameters. This can be used, for instance, to transition from straight to curly fur or to spatially vary its color.

In the following, we review the existing concept of neural radiance fields (Section 3.1), and then describe our approach for utilizing such fields as parametric primitives for building mesoscale appearance (Section 3.2), followed by a recipe for rendering their assemblies (Section 3.3). The optimization for recovering the reflectance field from images is discussed in Section 4.

3.1. Neural radiance fields

A radiance field is a function $f : \mathbb{R}^3 \times S^2 \to \mathbb{R}^3$ that maps a $5D$ spatio-directional scene coordinate $(\mathbf{x}, \omega)$, where *pos* is a location and ω is a direction, to the outgoing (RGB) radiance $L(\mathbf{x}, \omega)$. Mildenhall et al. [MST*20] approximate the true radiance field in a scene using a multi-layer perceptron f_θ, dubbed *neural radiance field*. They also extended the field to capture geometry information in the form of volumetric density $\sigma(x)$; the resulting mapping can be formally written as $f_\theta : (\mathbf{x}, \omega) \to (L(\mathbf{x}, \omega), \sigma(\mathbf{x}))$. Instrumenting the MLP to infer the density function σ is key to enable representing (and rendering) the scene without any additional geometric information. To render such extended fields, one needs to estimate the radiance reaching the image along each primary ray, expressed as

$$L(\mathbf{x}, \omega) = \int_0^\infty T(t)\, \sigma(\mathbf{x}_t)\, L(\mathbf{x}_t, \omega)\, \mathrm{d}t\,, \tag{1}$$

where $\mathbf{x}_t = \mathbf{x} - t\omega$ and $\mathbf{x}_s = \mathbf{x} - s\omega$ are points on the ray and $T(t) = \exp\left(-\int_0^t \sigma(\mathbf{x}_s)\mathrm{d}s\right)$ is the transmittance up to distance t.

Integration. In the original work, the authors propose to use two MLPs—a "coarse" one and a "fine" one—integrated using a two-stage quadrature rule. They begin by numerically integrating the coarse MLP and then refine the result using a second quadrature rule with adaptively spaced queries of the fine MLP. The integration is constrained to the interval between the near and far intersections of the ray with the camera frustum. Follow-up work has proposed alternative integration techniques [NSP*21, LMW21].

Network architecture. The MLP architecture utilized by Mildenhall et al. [MST*20] consists of several fully connected layers that are split into two main parts. The first stage processes only the positional coordinate $\mathbf{x}$, and outputs the view-independent density $\sigma(\mathbf{x})$. The second stage of the MLP takes the directional coordinate ω and a feature vector from the first stage and outputs the spatio-directional radiance $L(\mathbf{x}, \omega)$. We will refer to the first and the second stage as *spatial* and *spatio-directional* processing.

3.2. Parametric NeRF textures

We use a neural field as a building block for modeling mesoscale appearance. We wish to use distinct *reflectance* fields—NeRF textures[†]—for distinct mesoscale classes, such as fur, grass, fabric. However, the NeRF texture should be able to model diverse appearances within its class; i.e. the model should be parametric to allow changing the appearance. Furthermore, we want to enable filtering to combat aliasing artifacts when viewing the texture from a distance. Lastly, we need to enable relighting of a once-trained model to permit instantiating the NeRF texture over curved surfaces and in different orientations. Since each instance is expected to be relatively small, we will assume that all incident illumination reaching a single instance is due to a distant (parallel) light source.

We address all the aforementioned requirements by using a parametric neural field, represented by a single MLP $f_\theta : (\mathbf{x}, \omega, \omega_l, \beta) \to$

[†] We use the NeRF abbreviation to highlight the analogies to the original paper [MST*20], but we emphasize that the "R" stands for reflectance, not radiance, in our case.

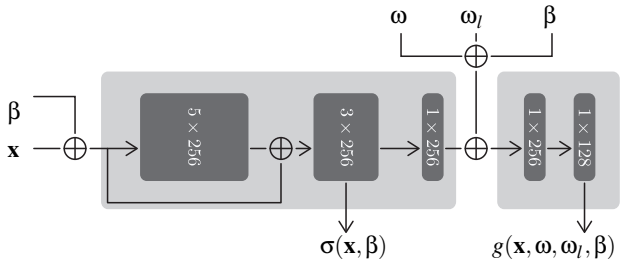

Figure 3: *We use the original NeRF architecture [MST*20] but condition it on additional appearance parameters β and lighting direction ω_l and use it to model the reflectance function g.*

$(\sigma(\mathbf{x},\beta), g(\mathbf{x},\omega,\omega_l,\beta))$, that models the volumetric density σ and a reflectance function g conditioned on the lighting direction ω_l and a set of filtering and appearance parameters β. The reflectance function $g(\mathbf{x},\omega,\omega_l,\beta)$ approximates the fraction of parallel illumination travelling in ω_l that hits a slab of the mesostructure, propagates through it, and leaves point $\mathbf{x}$ in direction ω after interacting with the volume density at $\mathbf{x}$; see Figure 2 where red polylines illustrate the transport that g accounts for. With the aforementioned field, the radiance collected along a ray reads:

$$L(\mathbf{x},\omega) = \int_0^\infty T(t)\sigma(\mathbf{x}_t,\beta) \int_S g(\mathbf{x}_t,\omega,\omega_l,\beta)\Phi(\omega_l)\,\mathrm{d}\omega_l\,\mathrm{d}t\,, \quad (2)$$

where $\Phi(\omega_l)$ is the radiant intensity of a distant emitter radiating in direction ω_l, and the inner integral integrates contributions of such emitters over the unit sphere of directions S.

Network inputs. We use the same architecture (and positional encoding) as proposed by Mildenhall et al. [MST*20], changing only the inputs to the network; see the illustration in Figure 3. The parameters β and ω_l are input to the MLP in the following way: the filtering radius and the appearance parameters that impact the density are concatenated to the inputs of the spatial stage. All other parameters in β are concatenated to ω and ω_l and input to the spatio-directional stage, which infers the reflectance function g.

The parameters β and ω_l are updated at each point where the MLP is queried to allow simulating spatially varying appearance and lighting, respectively. The parameter set β contains one parameter that controls the filtering of the mesostructure; the parameter is computed using ray differentials and allows the MLP to produce outputs with the appropriate level of detail. The remaining appearance parameters are specific to each mesostructure and can be modulated using artist-friendly means. We use classical surface textures mapping each query point to the nearest surface location and fetching β from the texture there.

Canonical training frame. Each unique NeRF texture is defined in a canonical coordinate frame. To allow instantiating it in the form of small, box-like elements, we bound the spatial extent of the mesostructure by an axis-aligned box. To place the element in the scene, we transform the canonical bounding box using an affine transformation. Spatial and directional coordinates of world-space NeRF queries are transformed into the canonical frame using the corresponding inverse transform.

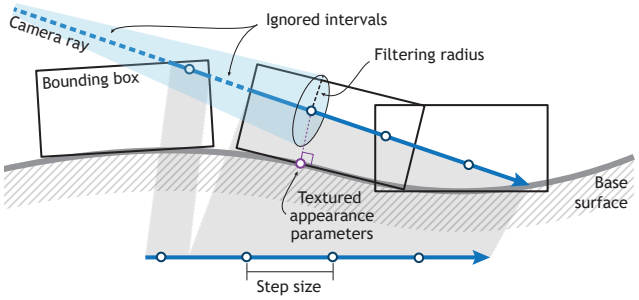

Figure 4: *We use a jittered equispaced ray marching that ignores empty intervals along the ray. When multiple NeRF instances overlap, we query one of them either stochastically or using a deterministic rule that aids the desired appearance.*

Texturing. We use a "base" mesh to define the overall shape of the object and an "anchor point" distribution on the surface of the mesh (Poisson disk distribution in most results) that defines the locations of the transformed origins of individual instances. The orientation of each instance is defined by the normal and tangent vectors of the base mesh at the anchor point; these correspond to the z and x axes of the canonical frame, respectively. The size of each instance is derived from an artist-defined parameter that allows roughly controlling the amount of overlap, which can be further modulated by the local density of points.

Using a base triangle mesh, a (non-uniform) point distribution, and classical texturing allows the artist to define high-level visual features of the mesostructure using familiar tools for sculpting and texturing classical meshes.

3.3. Ray marching

We use a straightforward quadrature rule with randomly offset, equispaced sample points along the ray [PKK00] to approximate the integrals in Equation (1). Our goal is to avoid querying the MLP in empty space. We constrain the marching to ray intervals that overlap with at least one of the instance bounding boxes and skip over empty intervals. For each ray, we first intersect all bounding boxes along the ray. Then we sort the entry and exit distances and extract an ordered list of intervals, where each interval stores a set of instances that it overlaps. During ray marching, we ignore the empty intervals such that the resulting steps would form an equispaced point sequence if the empty intervals were excluded (see Figure 4 for illustration). The handling of overlapping instances depends on how these should be interpreted; a discussion is provided in Appendix A.

Implementation details. To accelerate the computation of ray-box intersections, we put the bounding boxes of all NeRF instances into a single bounding volume hierarchy (BVH). The marching algorithm uses a constant step size that we adjusted manually to limit the amount of visible bias that the ray-marched estimation of transmittance introduces; alternative approaches are discussed in Section 6. At each step, we also compute a pixel-filter footprint using ray differentials [Ige99], which is input to the MLP as one of the appearance parameters β to permit filtered queries.

Figure 5: *Relighting a textured bunny by altering the global light direction. Since our models are conditioned on the light direction, we can relight objects easily by querying the network using light directions transformed to patch-local coordinates. The bunny consists of 3756 NeRF texture patches and was rendered at resolution 800x800 for approximately 30 seconds.*

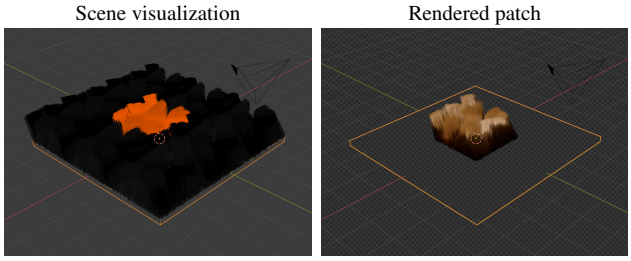

Figure 6: *To approximate shadowing and indirect illumination due to nearby patches at test time, we train on scenes where the rendered patch is surrounded by other instances with the same appearance (left). These additional instances are invisible to camera rays (right), but they absorb and scatter secondary rays.*

3.4. Lighting

We assume that each instance of the NeRF texture is small enough in the scene to warrant the assumption of distant light sources that cause parallel lighting. This simplifies the generation of training data for learning the reflectance function g. Nevertheless, a scene object can still be illuminated by nearby (point) lights as the lighting direction ω_l is recomputed at each marching step; the approximate reflectance function will be less accurate in such cases. Figure 5 shows the Stanford bunny with a plush-like, spatially varying appearance in three different lighting configurations. A single NeRF texture was used to produce all images.

In order to approximate direct illumination on the slab due to such light source with radiant intensity $\Phi(\omega_l)$, we cast a single shadow ray at each marching step in direction ω_l. The shadow ray is tested against all other objects in the scene, and would ideally march through each NeRF instance. This would, however, significantly increase the rendering cost. We thus opted for a cheaper approximation that intersects only the bounding boxes of NeRF primitives; see Appendix B for a discussion of artifacts.

We do not make any attempts to correctly synthesize long-distance global illumination in this article (short-distance GI is learned by the model), leaving the integration into a path tracer to future work. To mimic bouncing of light, we train the textures with a configurable amount of ambient lighting.

4. Optimization

We experimented with five NeRF textures: plush, fur, combed fur, grass, and carpet. We created a dataset for each using Blender and the Cycles path tracer, leveraging specialized plugins for fur and grass. The appearance of each texture can be adjusted after training using a subset of the plugin parameters β detailed below.

NeRF texture	β (excluding filtering parameter)	Figures
plush	2D: brightness, curliness	1, 5, 6, 10, 11, 15, 16
fur	2D: brightness, length	7, 8
combed fur	2D: brightness, fiber clumping	12
grass	1D: age (albedo & transparency)	7, 8, 9, 13, 14
carpet	3D: brightness of straight fibers, saturation and length of curly fibers	7, 8, 16

4.1. Data sets

Each dataset $\mathcal{D}$ contains images of one instance of a specific mesoscale class (e.g. a patch of fur). Each image is rendered from a different view and with randomized appearance parameters β and lighting that are uniformly sampled from predefined parameter ranges. Camera positions are sampled on an origin-centered hemisphere with the viewing vector pointing towards the origin. The patch is lit by a sphere-randomized directional light source.

To approximate shadowing and indirect illumination due to neighboring instances of a patch, we place eight additional instances around the center patch; see Figure 6. These instances are invisible to camera rays, but are taken into account when tracing the rest of the path. Unless specified otherwise, our datasets each contain 5000 training examples at resolution 512x512. Appendix C provides four example images from training sets for the fur and the carpet textures.

4.2. Procedure

We implemented the NeRF textures using Tensorflow [AAB*15] and integrated them into a custom Embree ray tracer [WWB*14]. For training, we used the Adam optimizer [KB15] with an initial learning rate of 5×10^{-4}, that is exponentially decayed with a rate of 0.1 over 500 000 batches. Each batch consists of 1024 radiance and transmittance estimates, which are obtained from four random images by ray marching along 1024 random rays cast from the reference cameras towards the bounding box of the patch. The optimization loss penalizes deviations in radiance using the symmetric mean absolute percentage error (SMAPE) and transmittance using the mean squared error (MSE), and it is computed by averaging the following per-ray expression over all the rays in the batch:

$$\mathcal{L}(i) = \frac{|L(i) - \widehat{L}(i)|}{L(i) + \widehat{L}(i) + \varepsilon} H(\widehat{T}(i)) + \left(T(i) - \widehat{T}(i)\right)^2, \qquad (3)$$

where $L(i)$ and $T(i)$ are the radiance and transmittance along the ray, respectively, $H(x)$ is the Heaviside step function that returns 1 for $x > 0$ and 0 otherwise, and $\varepsilon = 0.01$ is used to avoid a singularity in the denominator. $\widehat{L}$ and $\widehat{T}$ represent reference values that are obtained from the corresponding training image in the dataset. We found that using SMAPE instead of MSE leads to sharper detail reconstruction at the cost of slightly worse color reproduction.

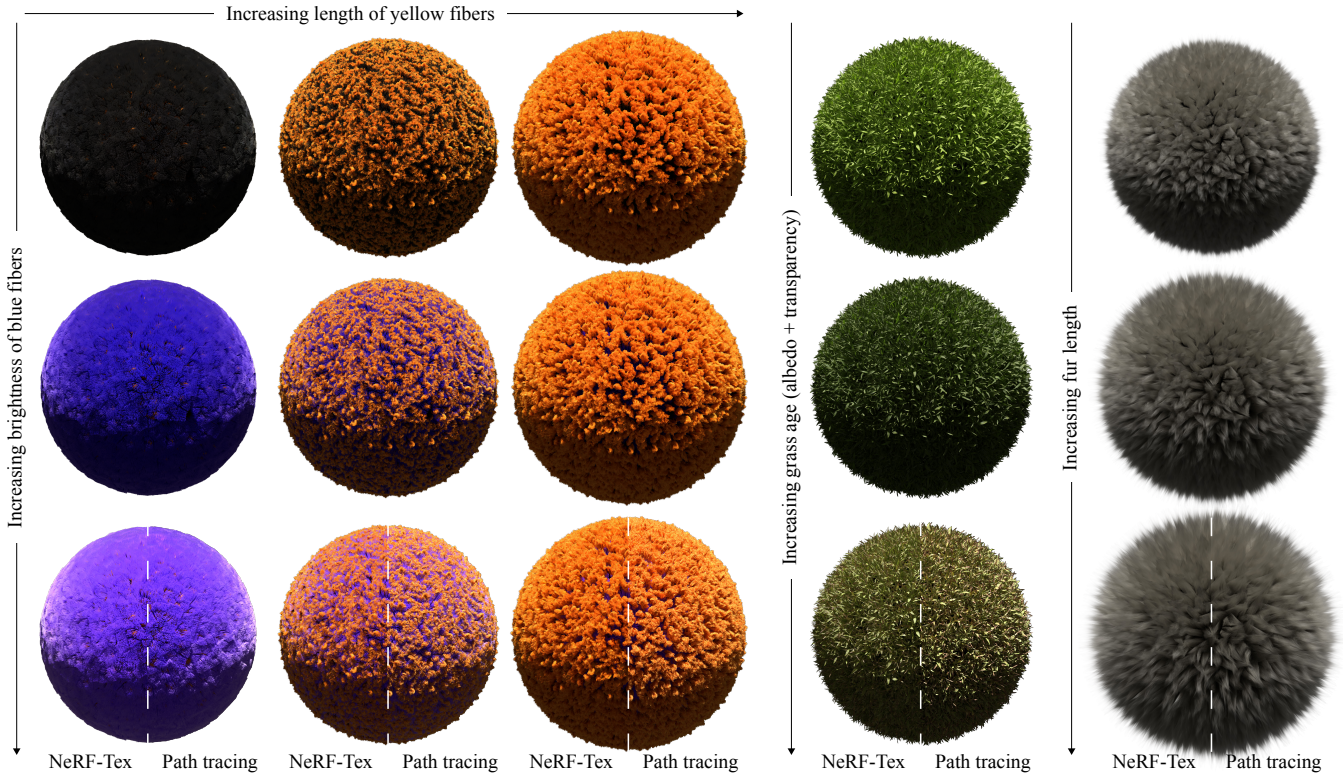

Figure 7: *Spheres textured with NeRF textures, each with a slightly different configuration of appearance parameters β. The right half of each image in the bottom row is rendered using reference path tracing to allow assessing the reproduction accuracy.*

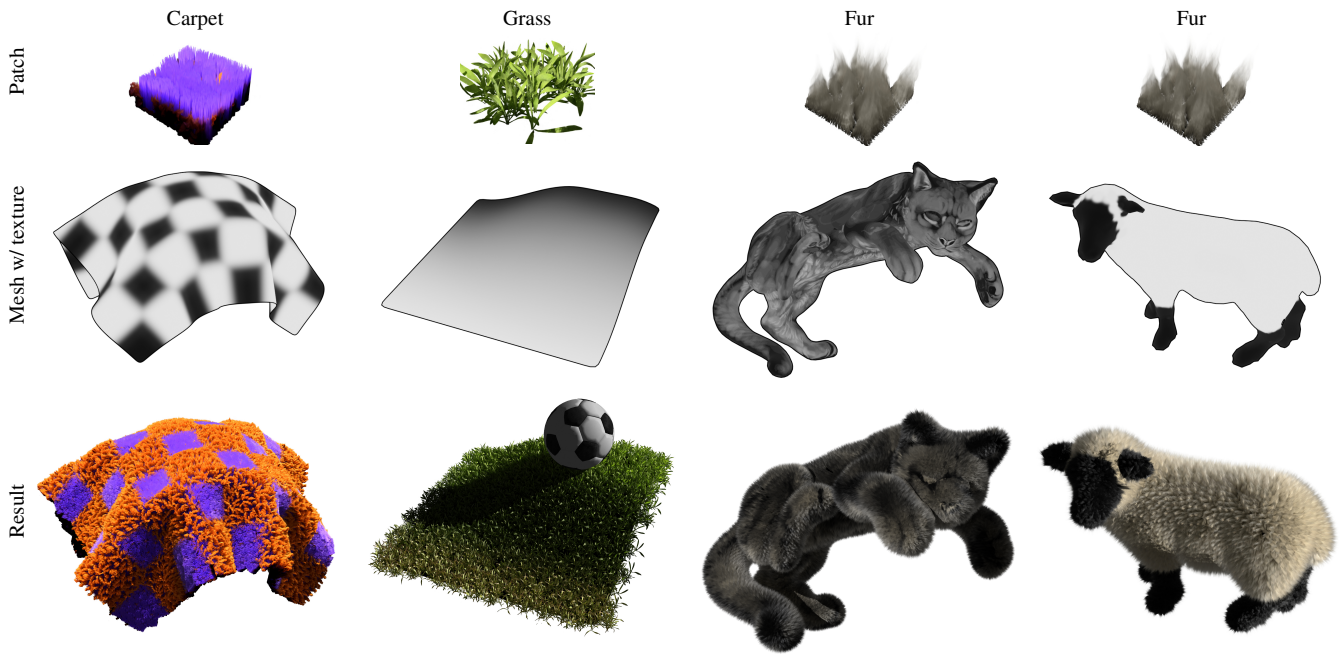

Figure 8: *Appearance can be modified spatially with the use of textures that modulate the appearance parameters β. In the first column the texture specifies the length of the yellow fibers, for the grass it controls its albedo and transparency, for the cat it adjusts brightness, and for the sheep it jointly modulates the length and brightness of the fur. The grass field also includes a soccer ball to show traditional meshes rendered with our NeRF textures.*

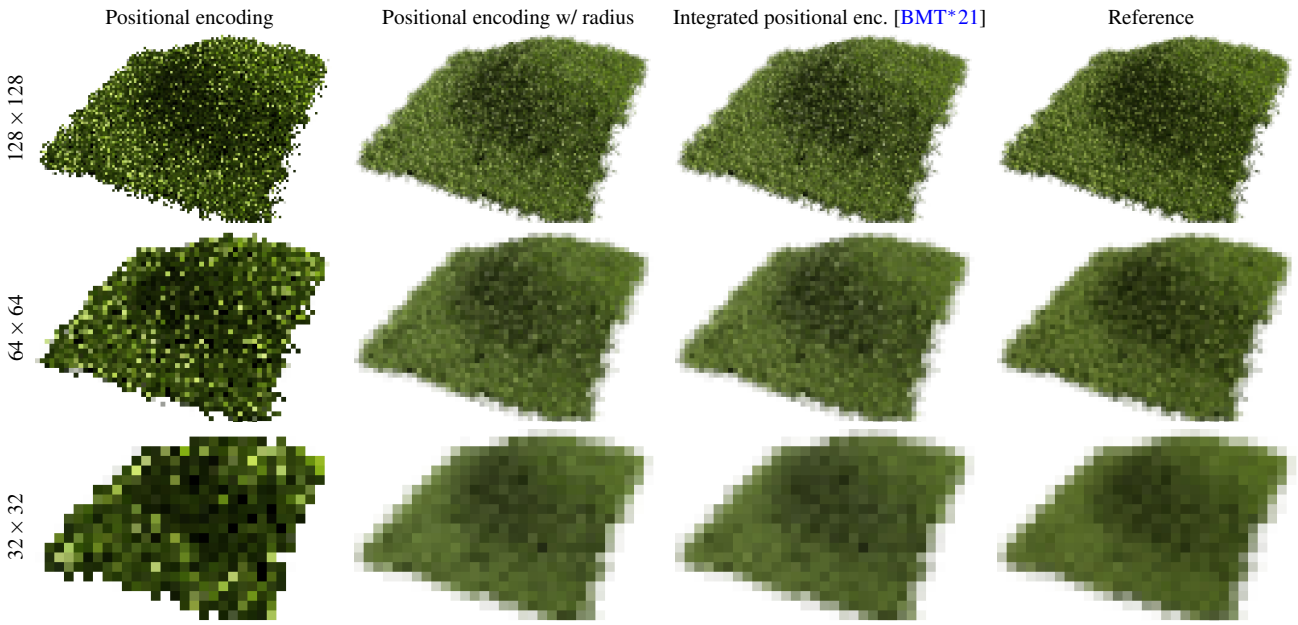

Figure 9: *NeRF textures permit filtered lookups. The first column shows a NeRF texture using the original positional encoding; aliasing artifacts appear. In the second and third column, we concatenate the cone radius to* β *or use the integrated positional encoding from mip-NeRF [BMT*21], respectively, and train the model with randomly blurred images in* β*. Both filtering approaches yield a good visual correspondence to the reference. All rows show the same scene but rendered with different image resolutions.*

5. Results

In this section we present renders of scenes with various base meshes that are textured with one of the NeRF primitives from Section 4.

Parametric appearance. Figure 7 and Figure 8 demonstrate the parametric nature of NeRF textures that allows synthesizing different appearances with one texture. The first three columns in Figure 7 show a 2D parameter sweep with the carpet texture, varying the brightness of straight (blue) fibers vertically, and varying the length of the curly fibers (yellow) horizontally. The subsequent columns show 1D parameter sweeps for the grass and one of the fur textures. Figure 8 contains objects with spatially varying surface textures that modulate the appearance of the NeRF textures. The supplemental video shows animated appearance parameters.

Filtering. Figure 9 demonstrates that NeRF textures facilitate filtered lookups. The scene consists of 176 instances of the grass texture. We compare the original positional encoding, which amounts to querying the MLP using point queries, to our approach of concatenating the radius of the ray cone at the query location to β, and to the integrated positional encoding by Barron et al. [BMT*21]. In all cases we train the model using blurred training images. In our method, we randomly blur each image and input the corresponding filtering radius at each point along the ray to the MLP as one of the β parameters. Both filtering approaches appear to yield similar visuals, although we note that the integrated positional encoding [BMT*21] may perform better in some configurations as it does not require conditioning the MLP on an extra parameter; this typically comes with lower reconstruction fidelity as analyzed in Figure 10.

Comparisons to original NeRF. We investigated the benefits and drawbacks of our design in comparisons to the original NeRF method [MST*20]. In Figure 10, we compare the original two-MLP NeRF model (a), our single MLP with ray marching constrained to the bounding box of the patch (b), and the same MLP but conditioned on parameters that control the lighting, appearance, and filtering. The parametric models in (c, d, e, f) yield lower accuracy on this specific patch. This is to be expected as these models allow relighting and (d, e, f) represent an entire space of appearances rather than a single specific instance. The reconstruction quality depends also on the size of the training dataset $\|\mathcal{D}\|$.

Figure 11 compares the quality when training a NeRF model on the whole object (top row) against our NeRF textures (middle row). As we model the entire mesoscale layer via instantiating a single learned NeRF primitive, the MLP focuses its capacity on a small, repetitive component of the scene. It is therefore not surprising that our approach captures fine details more accurately than the original method, which sets to the much harder problem of capturing the entire scene.

6. Analysis and discussion

This section discusses certain specifics and limitations of our approach and suggests potential improvements.

Patch placement. In our current implementation, each NeRF instance is transformed using an affine transformation. Since affine transformations preserve straight lines, the bounding boxes cannot closely follow a curved surface and might clip or overshoot

Reference	(a) [MST*20] coarse + fine MLPs, frustum traversal, MSE	(b) single MLP, bound. box traversal, SMAPE	(c) (b) + par. light.	(d) (c) + par. appearance, $\|\|\mathcal{D}\|\| = 5000$	(e) (d) w/ MSE instead of SMAPE	(f) (d) w/ $\|\|\mathcal{D}\|\| = 250$
SSIM ↑	0.929	0.947	0.919	0.870	0.853	0.846
LPIPS ↓	0.050	0.036	0.056	0.108	0.129	0.121
FLIP ↓	0.039	0.037	0.051	0.067	0.071	0.094

Figure 10: *The performance of the original two-stage NeRF approach (a) of Mildenhall et al. [MST*20] is worse compared to a single MLP and ray marching only within the bounding combined with our SMAPE based loss. The metrics (SSIM [ZBSS04], LPIPS [ZIE*18] and FLIP [ANAM*20]) were computed for multiple camera views and averaged. Models (d), (e) and (f) allow conditioning the brightness and curl of the fur on additional parameters; $\|\|\mathcal{D}\|\|$ is the size of the training set.*

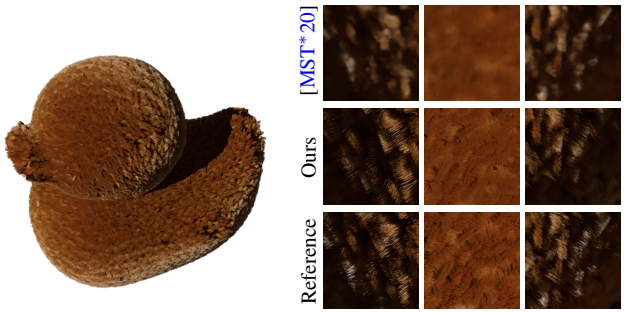

Figure 11: *The first row shows closeups of fur generated by the original NeRF model [MST*20] trained on the entire model of the duck. The second row shows the results achieved by our NeRF-Tex model, which uses patch instantiation instead.*

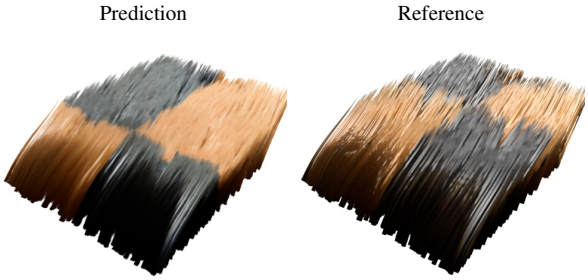

Figure 12: *We look up the appearance parameters based on the 3D query location projected onto the mesh. This can lead to artifacts as the appearance is assigned to a spatial location instead of individual hair strands, especially if the distance between the point projected onto the base mesh and the hair strand root is large.*

the surface of the base mesh. Nonrigid deformations could address this issue by allowing the patches to fit the surface curvature more accurately, similar to Neyret et al. [Ney98]. In such cases, the world-space rays correspond to curved lines in the canonical patch frame. Other NeRF works have proposed deforming a query location into another space to allow dynamic neural representations [PCP-MMN20, PSB*20]; these ideas could also be adapted to better align the NeRF textures with the surface. Shell maps [PBFJ05, JMW07] offer an alternative approach for applying NeRF textures to objects.

Visual Quality. In Figure 10 and Figure 12 we render individual NeRF textures next to path traced references. In contrast to earlier figures, where the mesoscale appearance at moderately distant views was fairly accurate, artifacts such as blurriness and splotchy appearance are easier to notice in closeup views. Concurrent works that further improve the reconstruction quality [ZRSK20, LGL*20, BMT*21, HSM*21], e.g. via better positional encodings [TSM*20, TLY*21], are likely to permit using our approach even for near views.

Performance. The ray marching technique described in Section 3.3, as well as other quadrature schemes proposed in prior and concurrent works, are fairly expensive due to the need to query the MLP at each step. The marching step size needs to be kept relatively small to maintain ray-marching bias comparable to artifacts of the MLP (see Figure 13). Unbiased transmittance estimators (e.g. delta, ratio, residual, and power-series estimators [Cra78, NSJ14, GMH*19]) are appealing as they could provide low-cost samples (albeit more noisy), if adapted to operate well on high-density, sparse structures—more research is needed to handle high-variation volumes efficiently. Another option for accelerating the rendering is to integrate the MLP over larger domains [BMT*21], e.g. entire rays [LMW21]. Traditional graphics approaches that importance sample non-point primitives [JNSJ11, NNDJ12, BJ17, SJ19] could serve as inspiration.

Parameter mapping. We use classical surface textures to specify the parameters conditioning the appearance of the NeRF texture. For a given point on the ray, the parameters are looked up from the texture at the nearest surface point of the base mesh. This leads to undesired artifacts for some mesostructures. An example with combed hair is shown in Figure 12, where the hair fibers wrongly change

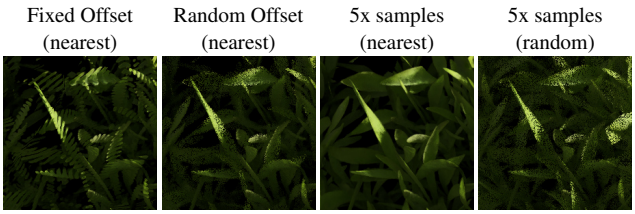

Figure 13: *Ray marching causes artifacts when the number of samples along a ray are too low. Fixed offsets lead to shapes partly disappearing whereas random offsets introduce noise. We can reduce the noise by taking additional samples. Sampling a random bounding box reduces clipping artifacts, but introduces more noise.*

appearance in certain regions. These artifacts could be reduced by learning a mapping between volume locations and points on the base mesh, e.g. in the spirit of deformable NeRF primitives [PCPMMN20, PSB*20], such that query positions map to strand roots. This could also allow additional modifications and deformations of learned textures.

Relation to traditional graphics approaches. Our current implementation is unlikely to rival the best combinations of traditional graphics approaches on static assets. For instance, combining the SGGX model [HDCD15] with an octree to fit a specific mesostructure will likely yield better and faster reconstruction (assuming the octree is well adapted to the geometry). However, extending such data structure to an entire space of appearances, e.g. from straight to curly fur, would require elaborate interpolation and domain specific approaches, which in turn reduces versatility. If developed further, we believe that neural material primitives will present a more practical alternative whenever training data for extracting a parametric appearance model is available.

Comparison to neural BTFs. Bidirectional texture functions (BTFs) are 6D functions capturing spatio-directional variations of appearance. Similar to radiance and reflectance fields, BTFs can also be efficiently represented using neural networks [RJGW19, RGJW20, KMX*21], but the two approaches excel in different situations. BTFs are more efficient for opaque materials, where ray marching tends to be computationally wasteful. The extra cost of ray marching is justified for volumetric, fuzzy materials, where multi-view consistency is easier to achieve through explicit volumetric integration, instead of having the network memorize integrals along all possible rays piercing through the mesostructure. Combining ray marching (NeRF-Tex) and single lookup methods (e.g. NeuMIP [KMX*21]) into a single neural material primitive that learns the optimal sampling strategy from data is interesting future work.

7. Future work

While our prototype lacks in certain respects (e.g. quality in closeup views, computation cost) the results are encouraging and we expect future work to further improve our method. In this section, we discuss the main challenges of integrating NeRF textures into path tracers and highlight two areas, where our approach can provide an edge over traditional graphics representations.

Integration into path tracing. Factoring out the lighting and utilizing the neural field as a reflectance function allows integrating NeRF textures into path tracing algorithms; provided that the assumption of distant lighting is acceptable. It remains to be investigated, however, whether our choice of representing transport using an MLP is optimal in such setups. Some alternative approaches represent only the phase function [GFWF20] (in addition to density), while others learn transmittances and simplified airlight integrals [SDZ*21]; the optimal approach is still actively sought. Furthermore, current NeRF approaches, including ours, march *through* the module(s) instead of learning the transport between points on the boundary, as proposed in many traditional modular approaches [LAM*11, LNJS12, ZHRB13, BNH*16]. This incurs higher cost: the MLP is evaluated multiple times, but it also factors out a large portion of the view dependency to facilitate accurate directional reconstruction. Additional research is needed, ideally in the context of production scenes, to devise the optimal aggregation strategy for neural fields.

Generative methods. Generative adversarial networks (GANs) [GPAM*14] have been extremely successful at synthesizing realistic natural imagery after training on large datasets [KLA*20]. Previous work by Schwarz et al. [SLNG20] and Chan et al. [CMK*20] show great promise in the application of generative adversarial concepts to neural radiance fields. Future work could use these ideas to create novel materials. This is especially fitting in the context of mesostructures as perfect reconstruction of a specific patch is not the ultimate goal; we would rather match only the relevant statistics and generate a brand new instance each time to break repetitive visuals.

Real-world capture. While all our NeRF textures were trained using synthetic datasets, our approach is readily applicable to datasets with natural images, as long as the intrinsic and extrinsic parameters of the camera and light sources are available. In fact, we deliberately preserved the original image-space loss (as opposed to computing the loss at points along the ray) to allow training NeRF textures on images of real materials in the future, captured using e.g. a light stage [DHT*00] or a collocated camera-light setup as proposed by Bi et al. [BXS*20]. Employing meta learning techniques, as explored by Sitzmann et al. [SCT*20] in the context of SDFs, could allow benefiting from both large synthetic and small captured image sets, pushing the boundaries of photorealistic rendering.

8. Conclusion

We proposed to repurpose neural radiance fields (NeRF) to learn a space of mesoscale appearances for materials such as grass and fur. The optimized volumetric primitives can be instantiated on surfaces to apply the materials to objects. We condition our networks on extra parameters that control the appearance and filtering, and which can be driven using textures. We showed that this approach provides a general method for learning complex materials that traditionally require a wide variety of tailored techniques. The versatility could be especially useful in the future for capturing diverse real-world materials. We believe our approach strikes a good balance between classical rendering and neural modeling, and, once extended to generative modeling, will further boost photorealistic image synthesis with classical rendering algorithms, such as path tracing.

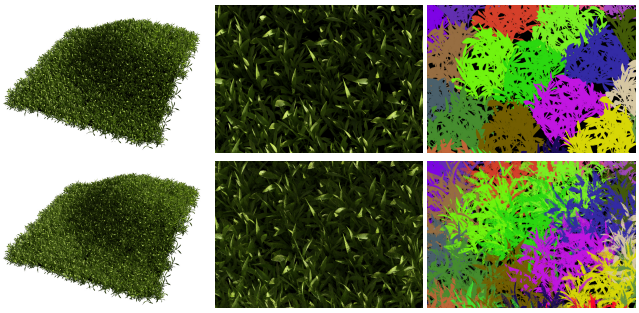

Figure 14: *Different approaches to handling query locations that are inside multiple bounding boxes lead to different visuals. The top row selects the bounding box nearest to the query location. The bottom row samples the bounding box randomly. The third column visualizes the selected patches. Images rendered w/ 1 spp.*

Acknowledgements

We thank Alex Evans and Towaki Takikawa as well as the anonymous reviewers for their helpful feedback. We also thank Virginia Ramp and Mary Langen for providing the mesh and texture of the cat model, respectively. Additionally, we used of the Bunny model from the Stanford 3D scanning repository.

Appendix A: Handling of overlapping instances

There are three options for handling multiple NeRF-Tex instances overlapping the ray marching location. First, the instances can be prioritized by an artist and the algorithm will pick the instance with the highest priority. Second, the contributions of individual instances can be added together and possibly weighted. If the weights add up to 1, then the volume density is not increased but the content combined by weighted averaging. If the the individual weights are set to 1, then the content of individual instances is added together. In the latter two cases, the instances can be queried stochastically at the cost variance; see Figure 14.

Appendix B: Shadow rays and transmittance estimation

We accelerate tracing of shadow rays by testing (binary) visibility only against bounding boxes of NeRF-Tex instances. Figure 15 illustrates the shadow artifacts that result from this simplified handling. Ideally, one would compute the fractional visibility, i.e. estimate the transmittance through the instances. This is unfortunately expensive. We thus opted for the simpler handling of shadows but note that recent transmittance estimators [GMH*19, KdPN21] could make the correct handling of fractional visibility viable.

Appendix C: Training data

Each NeRF texture is trained using a dataset with 5000 images that uniformly sample the appearance space that the texture should model. Example images from two different datasets are shown in Figure 16.

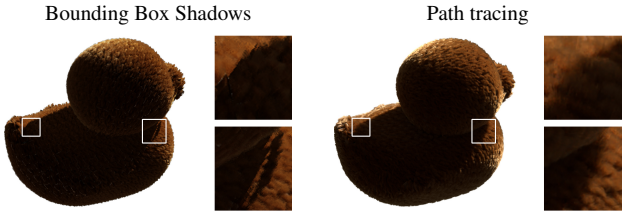

Figure 15: *Rendering artifacts may appear when accelerating lighting computations by testing shadow rays only against bounding boxes of individual patches. Casting a single shadow ray to approximate shadowing of the slab also leads to sharper shadows than in the path traced reference.*

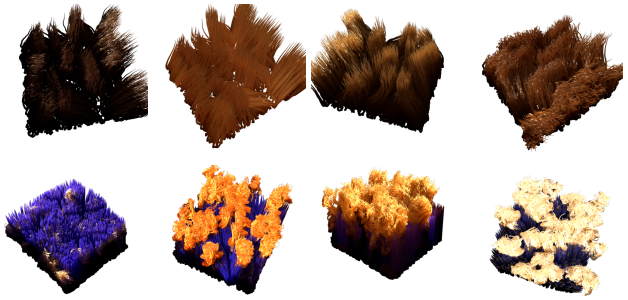

Figure 16: *Four target images from training sets that were used to create the fur (top) and the carpet (bottom) textures.*

References

[AAB*15] ABADI M., AGARWAL A., BARHAM P., BREVDO E., CHEN Z., ET AL.: TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. URL: https://www.tensorflow.org/. 5

[AB91] ADELSON E. H., BERGEN J. R.: The plenoptic function and the elements of early vision. In *Computational Models of Visual Processing* (1991), MIT Press, pp. 3–20. 2

[ACG*17] ALIAGA C., CASTILLO C., GUTIERREZ D., OTADUY M. A., LOPEZ-MORENO J., JARABO A.: An appearance model for textile fibers. *Computer Graphics Forum 36*, 4 (2017), 35–45. doi:10.1111/cgf.13222. 2

[ANAM*20] ANDERSSON P., NILSSON J., AKENINE-MÖLLER T., OSKARSSON M., ÅSTRÖM K., FAIRCHILD M. D.: FLIP: A difference evaluator for alternating images. *Proc. ACM Comput. Graph. Interact. Tech. 3*, 2 (Aug. 2020). doi:10.1145/3406183. 8

[BBJ*20] BOSS M., BRAUN R., JAMPANI V., BARRON J. T., LIU C., LENSCH H. P.: NeRD: Neural reflectance decomposition from image collections. *CoRR* (2020). 3

[BJ17] BITTERLI B., JAROSZ W.: Beyond points and beams: Higher-dimensional photon samples for volumetric light transport. *ACM Transactions on Graphics (Proceedings of SIGGRAPH) 36*, 4 (July 2017). doi:10.1145/3072959.3073698. 8

[BMT*21] BARRON J. T., MILDENHALL B., TANCIK M., HEDMAN P., MARTIN-BRUALLA R., SRINIVASAN P. P.: Mip-NeRF: A multiscale representation for anti-aliasing neural radiance fields. *arXiv* (2021). 1, 3, 7, 8

[BNH*16] BLUMER A., NOVÁK J., HABEL R., NOWROUZEZAHRAI D., JAROSZ W.: Reduced aggregate scattering operators for path tracing.

Computer Graphics Forum (Proceedings of Pacific Graphics) 35, 7 (Oct. 2016), 461–473. doi:10/f9c6w6. 9

[BRTO*21] B R M., TEWARI A., OH T.-H., WEYRICH T., BICKEL B., SEIDEL H.-P., PFISTER H., MATUSIK W., ELGHARIB M., THEOBALT C.: Monocular reconstruction of neural face reflectance fields. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2021). 3

[BXS*20] BI S., XU Z., SRINIVASAN P., MILDENHALL B., SUNKAVALLI K., HAŠAN M., HOLD-GEOFFROY Y., KRIEGMAN D., RAMAMOORTHI R.: Neural reflectance fields for appearance acquisition, 2020. arXiv:2008.03824. 1, 3, 9

[BZ17] BARNES C., ZHANG F.-L.: A survey of the state-of-the-art in patch-based synthesis. *Computational Visual Media 3*, 1 (Mar 2017), 3–20. doi:10.1007/s41095-016-0064-2. 2

[CBTB16] CHIANG M. J.-Y., BITTERLI B., TAPPAN C., BURLEY B.: A practical and controllable hair and fur model for production path tracing. *Computer Graphics Forum 35*, 2 (2016), 275–283. doi:10.1111/cgf.12830. 2

[CMK*20] CHAN E., MONTEIRO M., KELLNHOFER P., WU J., WETZSTEIN G.: pi-GAN: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *arXiv* (2020). 2, 9

[Cra78] CRAMER S. N.: Application of the fictitious scattering radiation transport model for deep-penetration monte carlo calculations. *Nuclear Science and Engineering 65*, 2 (1978), 237–253. doi:10.13182/NSE78-A27154. 8

[CRT20] CHAI M., REN J., TULYAKOV S.: Neural hair rendering. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XVIII* (2020), Vedaldi A., Bischof H., Brox T., Frahm J., (Eds.), vol. 12363 of *Lecture Notes in Computer Science*, Springer, pp. 371–388. doi:10.1007/978-3-030-58523-5_22. 2

[DHT*00] DEBEVEC P., HAWKINS T., TCHOU C., DUIKER H., SAROKIN W., SAGAR M.: Acquiring the reflectance field of a human face. *Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (2000). 9

[DI11] D'EON E., IRVING G.: A quantized-diffusion model for rendering translucent materials. *ACM Trans. Graph. 30*, 4 (July 2011). doi:10.1145/2010324.1964951. 2

[DN09] DECAUDIN P., NEYRET F.: Volumetric billboards. *Computer Graphics Forum 28*, 8 (2009), 2079–2089. doi:10.1111/j.1467-8659.2009.01354.x. 2

[DWd*08] DONNER C., WEYRICH T., D'EON E., RAMAMOORTHI R., RUSINKIEWICZ S.: A layered, heterogeneous reflectance model for acquiring and rendering human skin. *ACM Trans. Graph. 27*, 5 (Dec. 2008). doi:10.1145/1409060.1409093. 2

[DXT17] DESHMUKH P., XIE F., TABELLION E.: DreamWorks fabric shading model: From artist friendly to physically plausible. In *ACM SIGGRAPH 2017 Talks* (New York, NY, USA, 2017), SIGGRAPH '17, Association for Computing Machinery. doi:10.1145/3084363.3085024. 2

[GFWF20] GUO M., FATHI A., WU J., FUNKHOUSER T.: Object-centric neural scene rendering. *arXiv preprint arXiv:2012.08503* (2020). 1, 3, 9

[GMH*19] GEORGIEV I., MISSO Z., HACHISUKA T., NOWROUZEZAHRAI D., KŘIVÁNEK J., JAROSZ W.: Integral formulations of volumetric transmittance. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia) 38*, 6 (Nov. 2019). doi:10/dffn. 8, 10

[GPAM*14] GOODFELLOW I., POUGET-ABADIE J., MIRZA M., XU B., WARDE-FARLEY D., OZAIR S., COURVILLE A., BENGIO Y.: Generative adversarial nets. In *Advances in Neural Information Processing Systems* (2014), Ghahramani Z., Welling M., Cortes C., Lawrence N., Weinberger K. Q., (Eds.), vol. 27, Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf. 9

[GTB*12] GRAHAM P., TUNWATTANAPONG B., BUSCH J., YU X., JONES A., DEBEVEC P., GHOSH A.: Measurement-based synthesis of facial microgeometry. In *ACM SIGGRAPH 2012 Talks* (New York, NY, USA, 2012), SIGGRAPH '12, Association for Computing Machinery. doi:10.1145/2343045.2343057. 2

[HCJ13] HABEL R., CHRISTENSEN P. H., JAROSZ W.: Photon beam diffusion: A hybrid Monte Carlo method for subsurface scattering. *Computer Graphics Forum (Proceedings of EGSR) 32*, 4 (June 2013). doi:10/f445m4. 2

[HDCD15] HEITZ E., DUPUY J., CRASSIN C., DACHSBACHER C.: The SGGX microflake distribution. *ACM Trans. Graph. 34*, 4 (July 2015). doi:10.1145/2766988. 9

[HGC*20] HU B., GUO J., CHEN Y., LI M., GUO Y.: DeepBRDF: A deep representation for manipulating measured BRDF. *Computer Graphics Forum* (2020). doi:10.1111/cgf.13920. 2

[HMR20] HENZLER P., MITRA N. J., RITSCHEL T.: Learning a neural 3d texture space from 2d exemplars. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2020). 2

[HSM*21] HEDMAN P., SRINIVASAN P. P., MILDENHALL B., BARRON J. T., DEBEVEC P.: Baking neural radiance fields for real-time view synthesis. *arXiv* (2021). 1, 2, 8

[Ige99] IGEHY H.: Tracing ray differentials. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques* (USA, 1999), SIGGRAPH '99, ACM Press/Addison-Wesley Publishing Co., p. 179–186. doi:10.1145/311535.311555. 4

[JMLH01] JENSEN H. W., MARSCHNER S. R., LEVOY M., HANRAHAN P.: A practical model for subsurface light transport. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 2001), SIGGRAPH '01, Association for Computing Machinery, p. 511–518. doi:10.1145/383259.383319. 2

[JMW07] JESCHKE S., MANTLER S., WIMMER M.: Interactive smooth and curved shell mapping. In *Proceedings of the 18th Eurographics Conference on Rendering Techniques* (Goslar, DEU, 2007), EGSR'07, Eurographics Association, p. 351–360. 8

[JNSJ11] JAROSZ W., NOWROUZEZAHRAI D., SADEGHI I., JENSEN H. W.: A comprehensive theory of volumetric radiance estimation using photon points and beams. *ACM Transactions on Graphics (Presented at SIGGRAPH) 30*, 1 (Jan. 2011), 5:1–5:19. doi:10/fcdh2f. 8

[KB15] KINGMA D. P., BA J.: Adam: A method for stochastic optimization. In *ICLR (Poster)* (2015). URL: http://arxiv.org/abs/1412.6980. 5

[KCYM14] KONIARIS C., COSKER D., YANG X., MITCHELL K.: Texture mapping techniques for volumetric mesostructure. *Journal of Computer Graphics Techniques (JCGT) 3*, 1 (February 2014), 18–59. URL: http://jcgt.org/published/0003/01/02/. 2

[KdPN21] KETTUNEN M., D'EON E., PANTALEONI J., NOVÁK J.: An unbiased ray-marching transmittance estimator. *ACM Transactions on Graphics (Proceedings of SIGGRAPH) 40*, 4 (Aug. 2021). doi:10.1145/3450626.3459937. 10

[KK89] KAJIYA J. T., KAY T. L.: Rendering fur with three dimensional textures. *SIGGRAPH Comput. Graph. 23*, 3 (July 1989), 271–280. doi:10.1145/74334.74361. 2, 3

[KLA*20] KARRAS T., LAINE S., AITTALA M., HELLSTEN J., LEHTINEN J., AILA T.: Analyzing and improving the image quality of StyleGAN. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2020). 1, 9

[KMM*17] KALLWEIT S., MÜLLER T., MCWILLIAMS B., GROSS M., NOVÁK J.: Deep scattering: Rendering atmospheric clouds with radiance-predicting neural networks. *ACM Trans. Graph. (Proc. of Siggraph Asia) 36*, 6 (Nov. 2017). doi:10.1145/3130800.3130880. 2

[KMX*21] KUZNETSOV A., MULLIA K., XU Z., HAŠAN M., RAMAMOORTHI R.: NeuMIP: Multi-resolution neural materials. *ACM*

Transactions on Graphics (Proc. SIGGRAPH 2021) 40, 4 (2021). 1, 2, 3, 9

[LAM*11] LOOS B. J., ANTANI L., MITCHELL K., NOWROUZEZAHRAI D., JAROSZ W., SLOAN P.-P.: Modular radiance transfer. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia) 30*, 6 (Dec. 2011). doi:10/ddfbh9. 9

[LGL*20] LIU L., GU J., LIN K. Z., CHUA T.-S., THEOBALT C.: Neural sparse voxel fields. *NeurIPS* (2020). 1, 2, 8

[LH96] LEVOY M., HANRAHAN P.: Light field rendering. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1996), SIGGRAPH '96, Association for Computing Machinery, p. 31–42. doi:10.1145/237170.237199. 2

[LHW21] LEONARD L., HÖHLEIN K., WESTERMANN R.: Learning multiple-scattering solutions for sphere-tracing of volumetric subsurface effects. *Computer Graphics Forum (Proc. Eurographics) 40*, 2 (2021), 165–178. doi:10.1111/cgf.142623. 2

[LMW21] LINDELL D. B., MARTEL J. N. P., WETZSTEIN G.: AutoInt: Automatic integration for fast neural volume rendering. In *Proc. CVPR* (2021). 2, 3, 8

[LNJS12] LOOS B. J., NOWROUZEZAHRAI D., JAROSZ W., SLOAN P.-P.: Delta radiance transfer. In *Proceedings of ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, Mar. 2012), ACM. doi:10/gfzndh. 9

[LSS*19] LOMBARDI S., SIMON T., SARAGIH J., SCHWARTZ G., LEHRMANN A., SHEIKH Y.: Neural volumes. *ACM Transactions on Graphics 38*, 4 (Jul 2019), 1–14. doi:10.1145/3306346.3323020. 2

[LSS*21] LOMBARDI S., SIMON T., SCHWARTZ G., ZOLLHOEFER M., SHEIKH Y., SARAGIH J.: Mixture of Volumetric Primitives for Efficient Neural Rendering. arXiv:2103.01954. 1, 2

[MGZJ20] MONTAZERI Z., GAMMELMARK S. B., ZHAO S., JENSEN H. W.: A practical ply-based appearance model of woven fabrics. *ACM Trans. Graph. 39*, 6 (Nov. 2020). doi:10.1145/3414685.3417777. 2

[MJC*03] MARSCHNER S. R., JENSEN H. W., CAMMARANO M., WORLEY S., HANRAHAN P.: Light scattering from human hair fibers. *ACM Trans. Graph. 22*, 3 (July 2003), 780–791. doi:10.1145/882262.882345. 2

[MMR*19] MÜLLER T., MCWILLIAMS B., ROUSSELLE F., GROSS M., NOVÁK J.: Neural importance sampling. *ACM Trans. Graph. 38*, 5 (Oct. 2019), 145:1–145:19. doi:10.1145/3341156. 2

[MPG*16] MÜLLER T., PAPAS M., GROSS M., JAROSZ W., NOVÁK J.: Efficient rendering of heterogeneous polydisperse granular media. *ACM Trans. Graph. 35*, 6 (Nov. 2016), 168:1–168:14. doi:10.1145/2980179.2982429. 2

[MPH*15] MENG J., PAPAS M., HABEL R., DACHSBACHER C., MARSCHNER S., GROSS M., JAROSZ W.: Multi-scale modeling and rendering of granular materials. *ACM Transactions on Graphics (Proceedings of SIGGRAPH) 34*, 4 (July 2015). doi:10/gfzndr. 2

[MRKN20] MÜLLER T., ROUSSELLE F., KELLER A., NOVÁK J.: Neural control variates. *ACM Trans. Graph. 39*, 6 (Nov. 2020), 243:1–243:19. doi:10.1145/3414685.3417804. 2

[MST*20] MILDENHALL B., SRINIVASAN P. P., TANCIK M., BARRON J. T., RAMAMOORTHI R., NG R.: NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV* (2020). 1, 2, 3, 4, 7, 8

[Ney98] NEYRET F.: Modeling, animating, and rendering complex scenes using volumetric textures. *IEEE Transactions on Visualization and Computer Graphics 4*, 1 (1998), 55–70. doi:10.1109/2945.675652. 2, 3, 8

[NNDJ12] NOVÁK J., NOWROUZEZAHRAI D., DACHSBACHER C., JAROSZ W.: Virtual ray lights for rendering scenes with participating media. *ACM Transactions on Graphics (Proceedings of SIGGRAPH) 31*, 4 (July 2012). doi:10/gbbwk2. 8

[NSJ14] NOVÁK J., SELLE A., JAROSZ W.: Residual ratio tracking for estimating attenuation in participating media. *ACM Trans. Graph. 33*, 6 (2014), 179–1. doi:10.1145/2661229.2661292. 8

[NSP*21] NEFF T., STADLBAUER P., PARGER M., KURZ A., CHAITANYA C. R. A., KAPLANYAN A., STEINBERGER M.: DONeRF: Towards real-time rendering of neural radiance fields using depth oracle networks, 2021. arXiv:2103.03231. 3

[OMN*19] OECHSLE M., MESCHEDER L., NIEMEYER M., STRAUSS T., GEIGER A.: Texture fields: Learning texture representations in function space. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (October 2019). 2

[OMT*21] OST J., MANNAN F., THUEREY N., KNODT J., HEIDE F.: Neural scene graphs for dynamic scenes, 2021. arXiv:2011.10379. 1

[PBFJ05] PORUMBESCU S. D., BUDGE B., FENG L., JOY K. I.: Shell maps. In *ACM SIGGRAPH 2005 Papers* (New York, NY, USA, 2005), SIGGRAPH '05, Association for Computing Machinery, p. 626–633. doi:10.1145/1186822.1073239. 8

[PCPMMN20] PUMAROLA A., CORONA E., PONS-MOLL G., MORENO-NOGUER F.: D-NeRF: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020). 2, 8, 9

[PKK00] PAULY M., KOLLIG T., KELLER A.: Metropolis light transport for participating media. *Rendering Techniques 2000* (11 2000). doi:10.1007/978-3-7091-6303-0_2. 4

[PSB*20] PARK K., SINHA U., BARRON J. T., BOUAZIZ S., GOLDMAN D. B., SEITZ S. M., MARTIN-BRUALLA R.: Deformable neural radiance fields, 2020. arXiv:2011.12948. 2, 8, 9

[RGJW20] RAINER G., GHOSH A., JAKOB W., WEYRICH T.: Unified neural encoding of BTFs. *Computer Graphics Forum (Proc. Eurographics) 39*, 2 (July 2020), 167–178. doi:10.1111/cgf.13921. 2, 9

[RJGW19] RAINER G., JAKOB W., GHOSH A., WEYRICH T.: Neural BTF compression and interpolation. *Computer Graphics Forum (Proc. Eurographics) 38*, 2 (Mar. 2019), 235–244. doi:10.1111/cgf.13633. 2, 9

[RWG*13] REN P., WANG J., GONG M., LIN S., TONG X., GUO B.: Global illumination with radiance regression functions. *ACM Trans. Graph. 32*, 4 (July 2013). doi:10.1145/2461912.2462009. 2

[SCT*20] SITZMANN V., CHAN E., TUCKER R., SNAVELY N., WETZSTEIN G.: MetaSDF: Meta-learning signed distance functions. In *Advances in Neural Information Processing Systems* (2020), Larochelle H., Ranzato M., Hadsell R., Balcan M. F., Lin H., (Eds.), vol. 33, Curran Associates, Inc., pp. 10136–10147. URL: https://proceedings.neurips.cc/paper/2020/file/731c83db8d2ff01bdc000083fd3c3740-Paper.pdf. 9

[SDZ*21] SRINIVASAN P. P., DENG B., ZHANG X., TANCIK M., MILDENHALL B., BARRON J. T.: NeRV: Neural reflectance and visibility fields for relighting and view synthesis. In *CVPR* (2021). 1, 3, 9

[SJ19] SALESIN K., JAROSZ W.: Combining point and line samples for direct illumination. *Computer Graphics Forum (Proceedings of EGSR) 38*, 4 (July 2019), 159–169. doi:10/gf6rx6. 8

[SKU08] SZIRMAY-KALOS L., UMENHOFFER T.: Displacement mapping on the GPU — state of the art. *Computer Graphics Forum 27*, 6 (2008), 1567–1592. doi:10.1111/j.1467-8659.2007.01108.x. 2

[SKZ11] SCHRODER K., KLEIN R., ZINKE A.: A volumetric approach to predictive rendering of fabrics. *Computer Graphics Forum 30*, 4 (2011), 1277–1286. doi:10.1111/j.1467-8659.2011.01987.x. 2

[SLNG20] SCHWARZ K., LIAO Y., NIEMEYER M., GEIGER A.: GRAF: Generative radiance fields for 3d-aware image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)* (2020). 2, 9

[SRRW21] SZTRAJMAN A., RAINER G., RITSCHEL T., WEYRICH T.: Neural BRDF representation and importance sampling, 2021. arXiv:2102.05963. 2

[SZW19] SITZMANN V., ZOLLHÖFER M., WETZSTEIN G.: Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *Advances in Neural Information Processing Systems* (2019). 2

[TCC*20] TAN Z., CHAI M., CHEN D., LIAO J., CHU Q., YUAN L., TULYAKOV S., YU N.: MichiGAN: Multi-input-conditioned hair image generation for portrait editing. *ACM Trans. Graph. 39*, 4 (July 2020). doi:10.1145/3386569.3392488. 2

[TLY*21] TAKIKAWA T., LITALIEN J., YIN K., KREIS K., LOOP C., NOWROUZEZAHRAI D., JACOBSON A., MCGUIRE M., FIDLER S.: Neural geometric level of detail: Real-time rendering with implicit 3D shapes. 2, 3, 8

[TSM*20] TANCIK M., SRINIVASAN P. P., MILDENHALL B., FRIDOVICH-KEIL S., RAGHAVAN N., SINGHAL U., RAMAMOORTHI R., BARRON J. T., NG R.: Fourier features let networks learn high frequency functions in low dimensional domains. *NeurIPS* (2020). 8

[TZN19] THIES J., ZOLLHÖFER M., NIESSNER M.: Deferred neural rendering: Image synthesis using neural textures. *ACM Trans. Graph. 38*, 4 (July 2019). doi:10.1145/3306346.3323035. 1, 2

[VKJ19] VICINI D., KOLTUN V., JAKOB W.: A learned shape-adaptive subsurface scattering model. *ACM Trans. Graph. 38*, 4 (July 2019). doi:10.1145/3306346.3322974. 2

[WAA*00] WOOD D. N., AZUMA D. I., ALDINGER K., CURLESS B., DUCHAMP T., SALESIN D. H., STUETZLE W.: Surface light fields for 3d photography. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques* (USA, 2000), SIGGRAPH '00, ACM Press/Addison-Wesley Publishing Co., p. 287–296. doi:10.1145/344779.344925. 2

[WJV*05] WILBURN B., JOSHI N., VAISH V., TALVALA E.-V., ANTUNEZ E., BARTH A., ADAMS A., HOROWITZ M., LEVOY M.: High performance imaging using large camera arrays. *ACM Trans. Graph. 24*, 3 (July 2005), 765–776. doi:10.1145/1073204.1073259. 2

[WWB*14] WALD I., WOOP S., BENTHIN C., JOHNSON G. S., ERNST M.: Embree: A kernel framework for efficient cpu ray tracing. *ACM Trans. Graph. 33*, 4 (July 2014). doi:10.1145/2601097.2601199. 5

[YSJR17] YAN L.-Q., SUN W., JENSEN H. W., RAMAMOORTHI R.: A BSSRDF model for efficient rendering of fur with global illumination. *ACM Trans. Graph. 36*, 6 (Nov. 2017). doi:10.1145/3130800.3130802. 2

[YTJR15] YAN L.-Q., TSENG C.-W., JENSEN H. W., RAMAMOORTHI R.: Physically-accurate fur reflectance: Modeling, measurement and rendering. *ACM Trans. Graph. 34*, 6 (Oct. 2015). doi:10.1145/2816795.2818080. 2

[ZBSS04] ZHOU WANG, BOVIK A. C., SHEIKH H. R., SIMONCELLI E. P.: Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing 13*, 4 (2004), 600–612. doi:10.1109/TIP.2003.819861. 8

[ZHRB13] ZHAO S., HAŠAN M., RAMAMOORTHI R., BALA K.: Modular flux transfer: Efficient rendering of high-resolution volumes with repeated structures. *ACM Trans. Graph. 32*, 4 (July 2013). doi:10.1145/2461912.2461938. 2, 9

[ZIE*18] ZHANG R., ISOLA P., EFROS A. A., SHECHTMAN E., WANG O.: The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR* (2018). 8

[ZJMB11] ZHAO S., JAKOB W., MARSCHNER S., BALA K.: Building volumetric appearance models of fabric using micro CT imaging. *ACM Trans. Graph. 30*, 4 (July 2011). doi:10.1145/2010324.1964939. 2

[ZJMB12] ZHAO S., JAKOB W., MARSCHNER S., BALA K.: Structure-aware synthesis for predictive woven fabric appearance. *ACM Trans. Graph. 31*, 4 (July 2012). doi:10.1145/2185520.2185571. 2

[ZLB16] ZHAO S., LUAN F., BALA K.: Fitting procedural yarn models for realistic cloth rendering. *ACM Trans. Graph. 35*, 4 (July 2016). doi:10.1145/2897824.2925932. 2

[ZRSK20] ZHANG K., RIEGLER G., SNAVELY N., KOLTUN V.: NeRF++: analyzing and improving neural radiance fields, 2020. arXiv:2010.07492. 2, 8

[ZYWK08] ZINKE A., YUKSEL C., WEBER A., KEYSER J.: Dual scattering approximation for fast multiple scattering in hair. *ACM Trans. Graph. 27*, 3 (Aug. 2008), 1–10. doi:10.1145/1360612.1360631. 2