RESEARCH-ARTICLE

Generative Neural Materials

NITHIN RAGHAVAN, University of California, San Diego, San Diego, CA, United States

KRISHNA MULLIA, Adobe Inc., San Jose, CA, United States

ALEX TREVITHICK, University of California, San Diego, San Diego, CA, United States

FUJUN LUAN, Adobe Inc., San Jose, CA, United States

MILOŠ HAŠAN, Adobe Inc., San Jose, CA, United States

RAVI RAMAMOORTHI, University of California, San Diego, San Diego, CA, United States

Open Access Support provided by:

University of California, San Diego

Adobe Inc.

Published: 10 August 2025

Citation in BibTeX format

SIGGRAPH Conference Papers '25: Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers
August 10 - 14, 2025
BC, Vancouver, Canada

Conference Sponsors:
SIGGRAPH

Generative Neural Materials

NITHIN RAGHAVAN*, University of California San Diego, USA
KRISHNA MULLIA*, Adobe Research, USA
ALEXANDER TREVITHICK, University of California San Diego, USA
FUJUN LUAN, Adobe Research, USA
MILOŠ HAŠAN, Adobe Research, USA
RAVI RAMAMOORTHI, University of California San Diego, USA

Fig. 1. We present a conditional diffusion model for neural material generation. **Left:** An example scene with a number of neural materials in our representation, including training dataset materials, image-conditioned and text-conditioned generated materials. **Right:** Insets showing more detail and input conditions.

Advancements in neural rendering techniques have sparked renewed interest in neural materials, which are capable of representing bidirectional texture functions (BTFs) cheaply and with high quality. However, content creation in the neural material format is not straightforward. To address this limitation, we present the first image-conditioned diffusion model for neural materials, and show an extension to text conditioning. To achieve this, we make two main contributions: (1) we introduce a universal MLP variant of the NeuMIP architecture, defining a universal basis for neural materials as 16-channel feature textures, and (2) we train a conditional diffusion model for generating neural materials in this basis from flash images, natural images and text prompts. To achieve this, we also construct a new dataset of 150k neural materials in 16 categories, since no large-scale neural material data exists. To our knowledge, our work is the first to enable single-shot neural material generation from arbitrary text or image prompts.

CCS Concepts: • **Computing methodologies** → **Reflectance modeling**; **Neural networks**; **Learning latent representations**; **Texturing**; Image compression.

Additional Key Words and Phrases: neural materials, generative models, diffusion, text-to-material, image-to-material

ACM Reference Format:
Nithin Raghavan, Krishna Mullia, Alexander Trevithick, Fujun Luan, Miloš Hašan, and Ravi Ramamoorthi. 2025. Generative Neural Materials. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers (SIGGRAPH Conference Papers '25), August 10–14, 2025, Vancouver, BC, Canada.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3721238.3730746

1 Introduction

Traditional spatially-varying bidirectional reflectance distribution functions (SVBRDFs) with parameters such as base color, normal, roughness, specular color, etc., remain the most popular method for material representation due to their versatility, speed of evaluation and storage costs. However, real-world materials often do not obey the mathematical constraints imposed by analytic models such as the commonly used GGX and Beckmann microfacet distributions [Walter et al. 2007]. The texture of real materials is

*Both authors contributed equally to this research.

Authors' Contact Information: Nithin Raghavan, University of California San Diego, La Jolla, USA, n2raghavan@ucsd.edu; Krishna Mullia, Adobe Research, San Francisco, USA, mulliala@adobe.com; Alexander Trevithick, University of California San Diego, La Jolla, USA, atrevithick@ucsd.edu; Fujun Luan, Adobe Research, San Jose, USA, fluan@adobe.com; Miloš Hašan, Adobe Research, San Jose, USA, mihasan@adobe.com; Ravi Ramamoorthi, University of California San Diego, La Jolla, USA, ravir@cs.ucsd.edu.

SIGGRAPH Conference Papers '25, Vancouver, BC, Canada

ACM ISBN 979-8-4007-1540-2/25/08
https://doi.org/10.1145/3721238.3730746

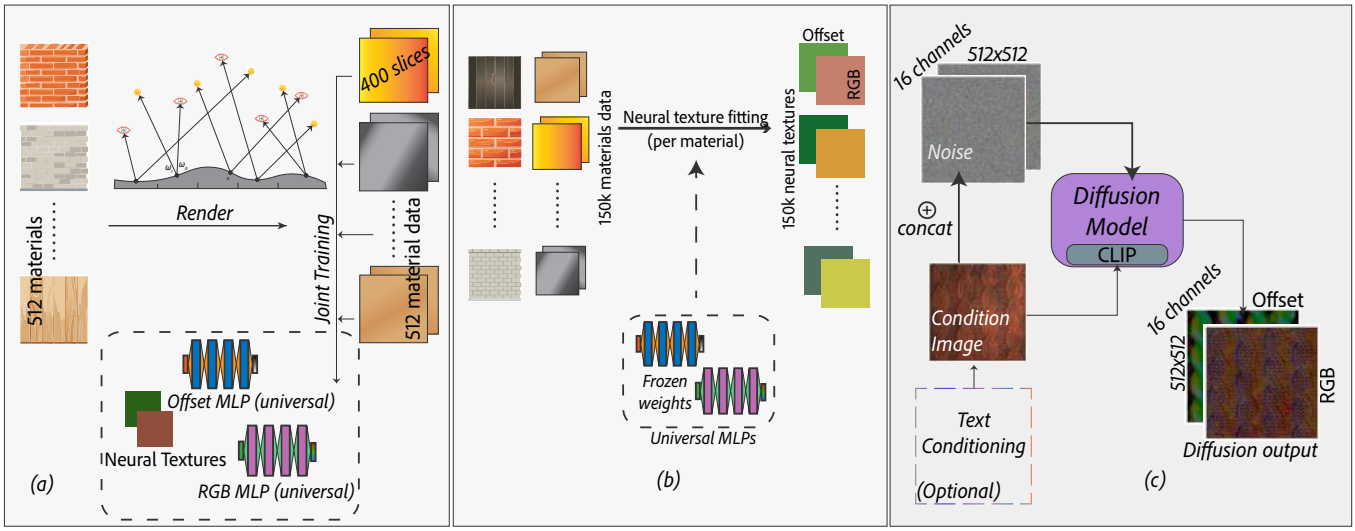

Fig. 2. **Overview of our pipeline.** (a) Shows the data generation and training of the universal basis network. For each of 512 selected materials, 400 slices of data are rendered, where each pixel contains a different camera and light direction. The 512 materials are jointly used to train (Offset and RGB) neural textures, as well as corresponding MLPs. (b) Shows the training process for the full dataset of approximately 150,000 materials. For each material, training data is rendered similarly as in (a), however, the universal MLPs are used with frozen weights to fit each material's neural texture. (c) The resulting 150,000 neural materials are used as training data for training the diffusion model.

commonly displaced, and often not even representable by height-fields. This structure causes shadowing and inter-reflection effects that SVBRDFs typically do not represent; instead, these phenomena need to be expensively simulated at the geometry level.

To address these limitations of SVBRDFs, bidirectional texture functions (BTFs) [Dana et al. 1999] were introduced to accurately represent materials with micro-scale and meso-scale surface irregularities without requiring explicit modeling of the local geometry. They represent a six-dimensional dictionary mapping from query directions to reflectance values, which, despite their accuracy, incur substantial memory costs. To alleviate this, recent neural material models [Kuznetsov et al. 2021, 2022; Rainer et al. 2022, 2019] have demonstrated remarkable success in efficiently compressing BTFs as nonlinear latent (feature) spaces coupled with a compact, per-material neural decoder. The inductive biases within these models enable the representation of parallax-mapped reflectance values, but per-material fitting requirements have presented challenges in scaling these methods. Fan et al. [2023] extended these approaches by introducing a universal neural decoder trained on a small set of 84 BTFs. However, they did not provide an approach for authoring new materials directly in the basis.

The generation of materials (unconditional, or conditioned on text, images, and sketches) has long been a topic of interest in the graphics literature, and several earlier works emphasized the importance of a generative, data-driven material prior for downstream tasks as well. Early efforts [Guo et al. 2020; Zhou et al. 2023, 2022] leveraged GAN-based approaches for SVBRDF generation. Recent years saw the adoption of diffusion-based models, which are now considered state-of-the-art; several methods [Sartor and Peers 2023; Vecchio et al. 2024a,b; Xue et al. 2024] have achieved impressive results in SVBRDF generation.

However, to our knowledge, few prior works have focused on the generation of neural materials [Diolatzis et al. 2023], and none leverage the power of large-scale diffusion models. This is due to several challenges: the lack of a unified neural material representation, the lack of a large enough training corpus, and the fact that most diffusion architectures are optimized for generating RGB data; they can be finetuned for mildly different modalities (albedo, normals, etc.) but neural material representations are substantially different from RGB images.

We solve these problems with the following contributions. First, given the current scarcity of neural material data, we present a new dataset of 150k neural materials in 16 families. Second, we construct a parallax-aware universal neural material basis. The latent (feature) space of this architecture results in a well-defined universal basis for neural materials, in the form of 16-channel neural feature textures, analogous to RGB channels in standard image generation.

Third, we further present a conditional diffusion model operating in the above 16-channel basis, trained on the new dataset, capable of single-shot neural material generation. The model can be conditioned on real photographs of materials under flash or natural lighting (or any other desired image inputs), and (by extension) on text prompts. In summary, our key contributions are the following:

- A universal MLP defining a universal basis for neural materials (Sec. 3.1)
- A conditional diffusion model for generating neural materials in this basis from flash images, natural images and text prompts (Sec. 3.2)

Both of the above models are trained on our dataset of 150k neural materials (Sec. 4.1). See also Fig. 2.

Our proposed pipeline enables full neuralization of the material pipeline from lightweight material sample capture (or text description) to final scene. Our representation seamlessly integrates into any renderer alongside classical materials (Fig. 1). We believe our work represents a step towards a scalable content creation pipeline for neural assets.

2 Related work

We first cover related methods on neural material representation and BTF compression, which generally focus on fitting, sampling and rendering (not generation). Next, we review existing generative models for materials.

2.1 Neural materials and BTF compressors

Many methods have been developed over the years to compress, interpolate and importance sample neural BTFs. We give a brief overview of these works below.

Autoencoder-based methods. Rainer et al. [2019] developed an asymmetric autoencoder to compress BTF slices into per-material latent vectors. This approach was later extended by Rainer et al. [2020] to a shared latent space; while this represents the first universal BTF network, it suffers from lack of representative capacity.

Decoder-only methods. Some methods eliminate an explicit encoder by treating each instance's latent code as a learnable parameter optimized jointly with a shared decoder [Tan and Mayrovouniotis 1995]. This framework was explored in recent works [Bojanowski et al. 2019; Park et al. 2019] and was adopted for neural materials in [Fan et al. 2023], described below.

NeuMIP. NeuMIP was the first to train coordinate-based neural networks for material regression, learning a high-resolution neural mipmap (texture pyramid) alongside the primary reflectance network to generalize spatial positional encodings and training on densely sampled synthetic BTF datasets. Unlike prior approaches, NeuMIP incorporated explicit inductive bias for parallax mapping through an *offset module* trained concurrently with the primary reflectance network, enabling parallax and masking effects without requiring modifications to the underlying geometry. Kuznetsov et al. [2022] added another module to fit silhouette effects. These methods are trained per material; the neural latent space is not reusable between materials.

Coordinate-based universal materials. Fan et al. [2022] define a neural BRDF feature space and a universal network on a large dataset of synthetic layered BRDFs and learned layering operators in this space, but the method is limited to SVBRDFs and has not been extended to BTFs. Fan et al. [2023] introduced the biplane feature space, representing a material as a pair of per-material neural textures (in the spatial and half-angle domain). They do not consider parallax and the space is trained on a very limited dataset of real BTFs. However, adding their half-angle space feature texture to our representation may be a useful future extension. NeuBTF [Rodriguez-Pardo et al. 2023] employed an asymmetric autoencoder to map input images to neural textures, allowing for convenient image-to-BTF mapping, but is constrained by the representational capacity of the latent space.

Other advancements. Xu et al. [2023] and Fu et al. [2024] introduce several efficient neural importance-sampling architectures for materials; this is an important orthogonal issue to material content creation, and could be applied to our representation in the future. Zeltner* et al. [2024] present a system similar to NeuMIP without a parallax module but with shading frame orientation prediction and importance sampling support.

2.2 Generative models for materials

GAN-based models. MaterialGAN [Guo et al. 2020], one of the earliest models for SVBRDF generation, adapted the StyleGAN2 architecture [Karras et al. 2019] and was trained on a synthetic SVBRDF dataset. It was also used as a prior to reconstruct SVBRDF maps from flash photographs. Building upon this approach, TileGen [Zhou et al. 2022] introduced improvements to enable tilability by imposing periodic boundary conditions within the convolutional layers. PhotoMat [Zhou et al. 2023] is a GAN-based material generation model trained on real flash photographs. This method does generate a limited neural material as an intermediate result, but the final generated materials are limited to SVBRDFs. Despite the promising results achieved by these methods, the inherent difficulties in training and scaling GAN-based models motivated a shift toward diffusion-based approaches.

Latent-space diffusion. Text2Mat [He et al. 2023] was among the first to train a variance-preserving diffusion process on a pretrained latent space of SVBRDFs, conditioned on CLIP-encoded text annotations. ControlMat [Vecchio et al. 2024a] replaced text embeddings with material embeddings rendered under global illumination, training on the entire Adobe Substance 3D Assets corpus [Adobe 2023]. This approach significantly improved results but relied on stitching multiple decoded patches to achieve higher resolution, introducing ghosting and blurring artifacts. MatFuse [Vecchio et al. 2024b] addressed these issues by incorporating diverse conditioning modalities, such as sketches, text, and images, to achieve resolutions up to 768×768. RGB↔X [Zeng et al. 2024] used image conditioning to obtain per-pixel material parameters for natural images (and vice versa), showing that simply concatenating the image condition with the diffusion input gives reasonable results. Alternatively, video diffusion models have been finetuned to estimate SVBRDF maps conditional on photo crops, reusing video frames as material maps [Ma et al. 2024]. Hyper-diffusion models have also been used to directly output the weights of an implicit BRDF [Gokbudak et al. 2024; Zhou et al. 2024].

Pixel-space diffusion. Recent advances have enabled high-resolution diffusion in pixel space, incorporating simplifications that make training more efficient [Crowson et al. 2024; Hoogeboom et al. 2023]. MatFusion [Sartor and Peers 2023], an early pixel-diffusion model, used a ControlNet-inspired [Zhang et al. 2023] conditioning mechanism to reconstruct SVBRDFs from images lit by collocated flash or natural lighting. ReflectanceFusion [Xue et al. 2024] adopted

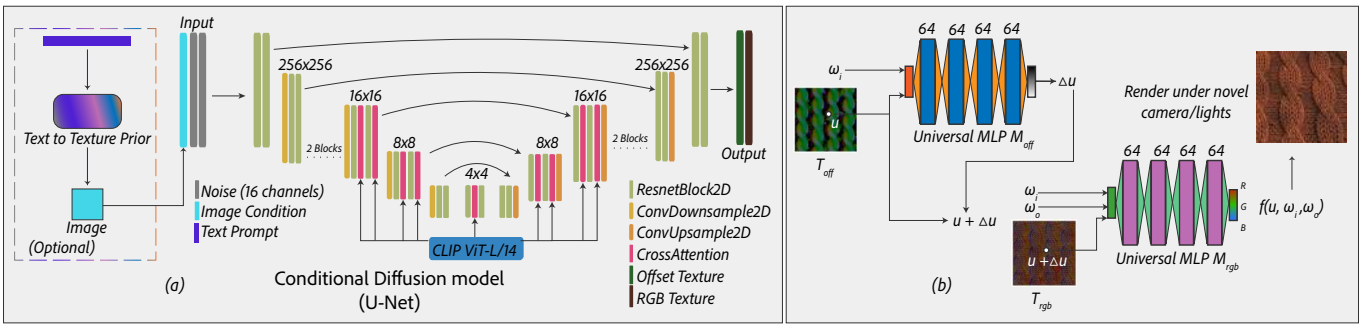

Fig. 3. **Network Architecture** (a) Shows our diffusion model's architecture, which can conditionally (image, or optionally text) generate a neural texture pair (offset and RGB). (b) Shows how the neural textures are used for inference, i.e., render under a given (novel) camera and light(s).

variance-preserving diffusion, achieving diverse SVBRDF generation with limited training data by training a residual network for Stable Diffusion XL [Podell et al. 2023] rather than creating a standalone model. In contrast, we propose a diffusion model for 16-channel learned neural materials.

3 Method

In this section, we introduce the key components of our solution:

(1) a universal MLP network extension of the NeuMIP neural material architecture [Kuznetsov et al. 2021] (Sec. 3.1); and
(2) a conditional diffusion model to generate neural material feature maps in this universal basis (Sec. 3.2).

The synthetic dataset of BTFs introduced in (Sec. 4.1) is used in the training of the universal MLPs and the diffusion model.

3.1 Universal basis network formulation

For simplicity, define a bidirectional texture function (BTF) as a six-dimensional function

$$f : [-1,1]^2 \times \mathbb{S}^2_+ \times \mathbb{S}^2_+ \to \mathbf{RGB} \tag{1}$$

Here $f(\boldsymbol{u}, \boldsymbol{\omega_i}, \boldsymbol{\omega_o})$ maps a location $\boldsymbol{u}$ in texture space, an incoming light direction $\boldsymbol{\omega_i}$ and an outgoing view direction $\boldsymbol{\omega_o}$ to a reflectance value. This definition is similar to a spatially varying BRDF, but the BTF can include effects such as parallax, occlusion, or interreflection, and will generally be non-reciprocal. NeuMIP [Kuznetsov et al. 2021] is an efficient neural architecture for BTF representation, combining two learned feature texture pyramids with two small per-material MLPs. We start from this architecture and omit the multi-resolution pyramid of NeuMIP, focusing on a single texture resolution for simplicity.

One issue is that each NeuMIP material has its own MLP weights in addition to feature textures; this representation is not straightforward to use as the output of a generative model. We found that making the MLP weights universal is sufficient for constructing a generalizable neural material basis, allowing us to directly generate feature textures. However, a key requirement is training the universal architecture on sufficiently large data. Our architecture has four components:

(1) a per-material offset feature texture T_{off} queried by bilinear interpolation at $\boldsymbol{u}$ to produce an offset feature vector $T_{\text{off}}[\boldsymbol{u}]$,
(2) a corresponding universal offset MLP M_{off} that explicitly incorporates parallax by estimating the view-dependent offset $\Delta\boldsymbol{u}$ to the original location $\boldsymbol{u}$, using the above feature vector,
(3) a per-material reflectance feature texture T_{rgb} interpolated at $\boldsymbol{u} + \Delta\boldsymbol{u}$ to produce a feature vector $T_{\text{rgb}}[\boldsymbol{u} + \Delta\boldsymbol{u}]$;
(4) a universal reflectance network M_{rgb} capable of decoding the above feature vector, $\boldsymbol{\omega_i}$ and $\boldsymbol{\omega_o}$ to an RGB reflectance.

The final reflectance value $\hat{f}_{\phi,\psi}(\boldsymbol{u}, \boldsymbol{\omega_i}, \boldsymbol{\omega_o})$ is computed as follows, where ϕ denotes the collective weights of M_{rgb} and M_{off} and ψ denotes the texel values of T_{rgb} and T_{off}:

$$\Delta u = M_{\text{off}}(T_{\text{off}}[\boldsymbol{u}], \boldsymbol{\omega_o}) \qquad \in [-1,1]^2, \tag{2}$$

$$\hat{f}_{\phi,\psi}(\boldsymbol{u}, \boldsymbol{\omega_i}, \boldsymbol{\omega_o}) = M_{\text{rgb}}(T_{\text{rgb}}[\boldsymbol{u} + \Delta\boldsymbol{u}], \boldsymbol{\omega_i}, \boldsymbol{\omega_o}) \in \mathbf{RGB}. \tag{3}$$

Our universal architecture (and the pipelines for its training and evaluation) can be found in Figure 2. The key consequence of this architecture (once trained on a sufficiently large material dataset) is the existence of a *single* basis for all neural materials, and a generative model can be trained to simply output textures T_{off} and T_{rgb} in this basis. We demonstrate that a wide variety of diffuse and specular material properties as well as complex microgeometry and mesoscale patterns can be represented in this basis in Figure 4. More examples can be found in the supplementary Fig. S5.

3.2 Diffusion formulation

Once we have defined the universal MLPs and the corresponding basis coefficients for neural materials in the form of feature textures T_{off} and T_{rgb}, the problem of generating neural materials can be reduced to training a diffusion model generating these feature textures in the form of multi-channel images.

Latent diffusion models, such as those proposed by Rombach et al. [2022] , have demonstrated impressive results at high resolutions. However, these models typically require a large and diverse set of exemplars to effectively learn a valid latent space representation.

We therefore opt for pixel diffusion and build upon the advancements presented in Simple Diffusion [Hoogeboom et al. 2023], which demonstrate that convergence at higher resolutions can be significantly improved through a small number of modifications to the standard training procedure. These are described in Sec. 4. A full visualization of our model can be found in Figure 3 (a).

Image conditioning. We condition the diffusion model on RGB images of materials under natural or flash lighting. This allows a user to use lightweight capture of material samples, or use any suitable image, to generate a neural material resembling the image prompt. During training, given an input material, we render it onto a plane, illuminated by either a point light or one of 48 randomly rotated indoor and outdoor environment maps from the HDRIHaven repository [PolyHaven 2025]. The normalized render is concatenated with the diffusion model inputs, and also fed through a CLIP ViT-L/14 head [Radford et al. 2021] into the cross-attention conditioning inputs of the model. Both conditions are simultaneously dropped for 10% of samples, to support unconditional generation and classifier-free guidance.

Text conditioning. The generalizability of our model allows for material generation from text through the use of an intermediate tileable texture generator [Aggarwal et al. 2023]. Figure 2 (d) illustrates the integration of image and text conditioning into the diffusion model. As seen in Figure. 5, our model can incorporate such images effectively despite being trained on synthetic data.

Architecture. We implement our denoiser as a velocity prediction network with a UNet architecture [Ronneberger et al. 2015]; the native resolution is 512×512. The architecture is illustrated in Figure 2 (a). We use the Diffusers API [von Platen et al. 2022] to construct a UNet with default settings, two layers per block and `block_out_channels` = [128, 128, 256, 512, 512, 1024, 1024]. We base our training code on Simple Diffusion [Favero 2024; Hoogeboom et al. 2023]. All blocks are `DownBlock2D`, except for the third-to-last and second-to-last blocks, which are `CrossAttnDownBlock2D`. We use periodic (circular) boundary conditions for all convolutional layers to enforce tileability of the resulting feature maps.

4 Dataset Rendering and Model Training Details

In this section, we cover the implementation details of our method, in this order:

(1) Sourcing and transforming the synthetic data used to train the universal basis (Sec. 4.1);
(2) Training M_{rgb} and M_{off} (Sec. 4.2);
(3) Fitting T_{rgb} and T_{off} for our full BTF dataset (Sec. 4.3); and
(4) Diffusion training details (Sec. 4.4).

4.1 Rendering the BTF Dataset

In this section, we describe the process of rendering our BTF data from a collection of SVBRDFs. This data is subsequently fit into a collection of 148, 246 neural materials with a resolution of 512×512 (ref. Sec 4.3).

Synthetic data. Since no sufficiently large BTF repository exists, we generate the required data from SVBRDFs sourced from the Adobe Substance 3D Assets library [Adobe 2023]. Our materials are categorized into sixteen material types: ceramic, concrete/asphalt, fabric, ground, leather, marble/granite, metal, organic, paint, paper, plaster, plastic/rubber, stone, terracotta, wood (outdoor), and wood. These SVBRDFs consist of five material maps at 2048×2048 resolution: base color, normal, height, roughness and metallic.

Scene setup. We set a BTF resolution of 512×512 texels and generate $N_s = 400$ pairs of random view/light configurations per texel. A densely tessellated plane (2048×2048) with height displacement (to incorporate complex meso-scale effects such as interreflections, self-shadowing, subsurface scattering) is procedurally generated and instanced in a 3×3 grid to preserve tileability. The resulting synthetic BTFs also include anti-aliasing effects.

Rendering. Using a custom modification of the Blender Cycles renderer, each texel and each view/light pair is rendered independently using 128 path samples. For each view/light pair, a ray is sampled towards the texel footprint along the direction $-\boldsymbol{\omega_o}$. A sun light with an angular diameter of 2° is positioned along the light direction ω_i, and its intensity is normalized to maintain unit irradiance. Each material takes approximately 3.75 minutes to load, render and upload to storage on an Amazon EC2 instance (4 NVIDIA T4 GPUs, 48 CPUs); we parallelize the rendering across several instances. The resulting data is around 2GB per material.

4.2 Training M_{rgb} and M_{off}

In this stage, we simultaneously optimize the universal networks M_{rgb} and M_{off} on 512 selected materials, as well as their feature textures. Our goal is to learn shared networks that work across several material families and parallax levels. The textures are discarded after training; they can be refit using the procedure described in Sec. 4.3.

Initial texture selection. We choose 32 of the most-displaced materials from each of our 16 families to form our training set of $N_m = 512$ materials, representing the hardest examples. We observe that more materials do not significantly impact the result quality (Table 1).

Universal MLP training scheme. Each of the 512 texture weights are sampled from a normal distribution of variance 0.01, and the network weights from a normal distribution with variance 0.1. During each iteration, we select 2 slices of 512×512 texels from each of 8 randomly-chosen materials (out of the full set of 512 materials selected for universal MLP training). We set the learning rate for the networks to 3×10^{-4} and for the textures to 10^{-3} and train for 100 epochs on eight 40 GB NVIDIA A100 GPUs, totalling 9.75 hours.

Prefiltering feature textures. Similar to NeuMIP, independently optimizing the neural feature texels introduces noise and slows convergence; to address this, we follow a coarse-to-fine approach: we initialize two Gaussian kernels with standard deviations $\tau_{\text{RGB}}(t)$, $\tau_{\text{off}}(t)$ and prefilter each texture prior to bilinear interpolation in every training iteration. Their time-evolution follows

$$\tau(t) = \max(\text{INIT} \cdot 2^{-t/h}, \text{FINAL})$$

For τ_{RGB} we set $h = 3847$, INIT = 8 and FINAL = $\frac{1}{2}$, while for τ_{off} we set $h = 3847$, INIT = 8 and FINAL = 2.

4.3 Fitting T_{rgb} and T_{off}

In this stage, we obtain a feature texture (a set of basis coefficients) for each material in the BTF dataset.

Per-material fitting scheme. For both T_{rgb} and T_{off}, we define the latent vector size to be 8 and the resolution to be 512×512 to match the BTF resolution. Both textures are sampled initially from a Gaussian of variance 0.01. We freeze the weights of both universal networks and optimize the feature textures with a learning rate of 10^{-2} until the training PSNR converges. These materials can differ significantly in complexity, leading to wildly differing training requirements and initial fitting conditions. As a result, we heuristically determine the parameters $\textsc{init} = 20, h = 500, \textsc{final} = 0.5$ for τ_{RGB} and $\textsc{init} = 20, h = 500, \textsc{final} = 2$ for τ_{off}. We then optimize the following statement for 2000 iterations with a batch size of 4 512×512 slices per iteration, optimizing for the texture weights ψ while holding the network weights ϕ fixed:

$$\arg\min_{\psi} \mathop{\mathbf{E}}_{\boldsymbol{\omega_o}, \boldsymbol{\omega_i} \sim \mathcal{U}(\mathbb{S}^2_+)} \mathop{\mathbf{E}}_{\boldsymbol{u} \sim \mathcal{U}(0,1)^2} \|\hat{f}_{\phi,\psi}(\boldsymbol{u}, \boldsymbol{\omega_i}, \boldsymbol{\omega_o}) - f(\boldsymbol{u}, \boldsymbol{\omega_i}, \boldsymbol{\omega_o})\|_1$$

This process takes around 9.5 minutes per material on an Amazon EC2 instance (1 Nvidia T4 GPU, 8 CPUs), and we parallelize the fitting across multiple instances like before. After fitting, we blur each texture with its corresponding FINAL-value and stack the two textures to produce a $512 \times 512 \times 16$ dimensional set of per-material coefficients in the neural basis.

4.4 Diffusion training details

In this stage, we train the diffusion model on the dataset of $(T_{\text{rgb}}, T_{\text{off}})$ feature texture pairs obtained in the previous section. We adopt a variance-preserving noise schedule, in which the signal- and noise-scaling coefficients $\alpha(t)$ and $\sigma(t)$ are defined so as to satisfy $\alpha^2(t)^2 + \sigma^2(t) = 1$.

We set the batch size to 512 and the learning rate to 9.5×10^{-4}. Our 512×512 model took 18 days to train on eight 40 GB NVIDIA A100 GPUs. After training, any sampling scheme (DDIM, DDPM, etc.) can be used to generate materials from the network.

During training, we incorporate several modifications inspired by Hoogeboom et al. [2023] and Hang et al. [2024] to help convergence of our network at high resolutions. These are listed below:

Noise schedule shifting. Given that $\alpha(t)$ and $\sigma(t)$ satisfy $\alpha^2(t) + \sigma^2(t) = 1$, the cosine noise schedule defined by

$$\textsc{snr}(t) = \frac{\alpha(t)}{\sigma(t)} = \tan\left(\frac{\pi t}{2}\right)$$

generates both functions. At high resolutions, not enough noise is introduced for small values of t, so we attenuate the noise schedule by an additional factor, where $H > 64$:

$$\textsc{snr}_{\text{shift}}(t) = \tan\left(\frac{\pi t}{2}\right) \cdot \left(\frac{64}{H}\right)^2$$

Min-SNR weighting. To further speed convergence, we implement the Min-SNR weighting strategy from [Hang et al. 2024] by setting $w(t) = (1 + \textsc{snr}(t))^{-1}$.

Nonlinear scaling. We empirically observed that denoising a nonlinearly scaled transformation of the input further enhances convergence; our full pipeline applies the inverse operation after sampling and prior to rendering. As most of our neural material data lies within the range $[-1, 1]$ we found the cube root function effective for this purpose.

Multiscale noise downsampling. To emphasize convergence on low-frequency details, we downsample the noise (obtained from our predicted velocity) for each element in a sequence of resolutions $\{H_i\} = \{2^i\}_{i=6}^{\log H}$, compute the loss for each H_i and weight them by their corresponding resolutions before aggregating them. Our final diffusion loss function is then

$$\ell_{\text{diff}}(\hat{\epsilon}, \epsilon) = w(t) \sum_i \frac{64}{H_i} \|(D(\hat{\epsilon}_t, H_i) - D(\epsilon_t, H_i))\|_2^2 \tag{4}$$

where $D(\cdot, \cdot)$ is the downsampling operation.

5 Results

In this section, we first justify the choice of architecture for our universal basis, and then demonstrate the generalization of our learned universal basis by showcasing several materials from the dataset rendered in 4.1 fitted to our network. We then demonstrate results from our 512×512 resolution model on four image categories: held-out validation materials, real-world flash captures from the PhotoMat dataset [Zhou et al. 2023], real-world phone captures under unknown illumination, and finally text-to-material results using a text-to-texture prior model [Aggarwal et al. 2023] which produces a tileable image input for our model from a text prompt. We visually compare our outputs with that of PhotoMat, briefly discuss the tilability of our model, and provide performance metrics for both rendering and sampling from the dataset. Finally, we go over limitations of our current framework. Results from our 256×256 model can be found in the supplementary. We additionally urge readers to view the supplementary videos, as they provide a much better demonstration of the parallax effects seen during changing view.

5.1 Choice of universal architecture

We validate the choice of network size in Table 1 by comparing model sizes and training durations on a validation set of 256 materials. The validation set is generated using the same parameters as the full training set, with 16 materials per category. Given the comprehensive validation requirements, which involve both generalization during fitting and subsequent prediction, we evaluate performance by fitting each material using half of the training data and measuring the PSNR on the remaining 200 materials.

Our primary objective is minimizing the network size, as the network must also generate the image conditions for our diffusion model during each iteration, as well as integrating quickly into standard renderers. The importance of smaller widths is illustrated in the timings recorded in Table S1, where the speed of larger-width networks drastically drops as the width is increased due to register pressure. The importance of smaller depths is shown in Figure S4, where larger-depth networks are observed to lose color and shadow details relative to the ground truth when trained for equal iterations. This quality drop with larger networks is likely due to the lack of skip connections and normalization, used in heavier architectures such as those by Fan et al. [2022] and Fan et al. [2023], which we avoided for rendering performance reasons.

Therefore, despite the potential for larger networks to achieve higher PSNRs with extended architectures and training, we choose a depth of 4 and a width of 64 for both M_{rgb} and M_{off}.

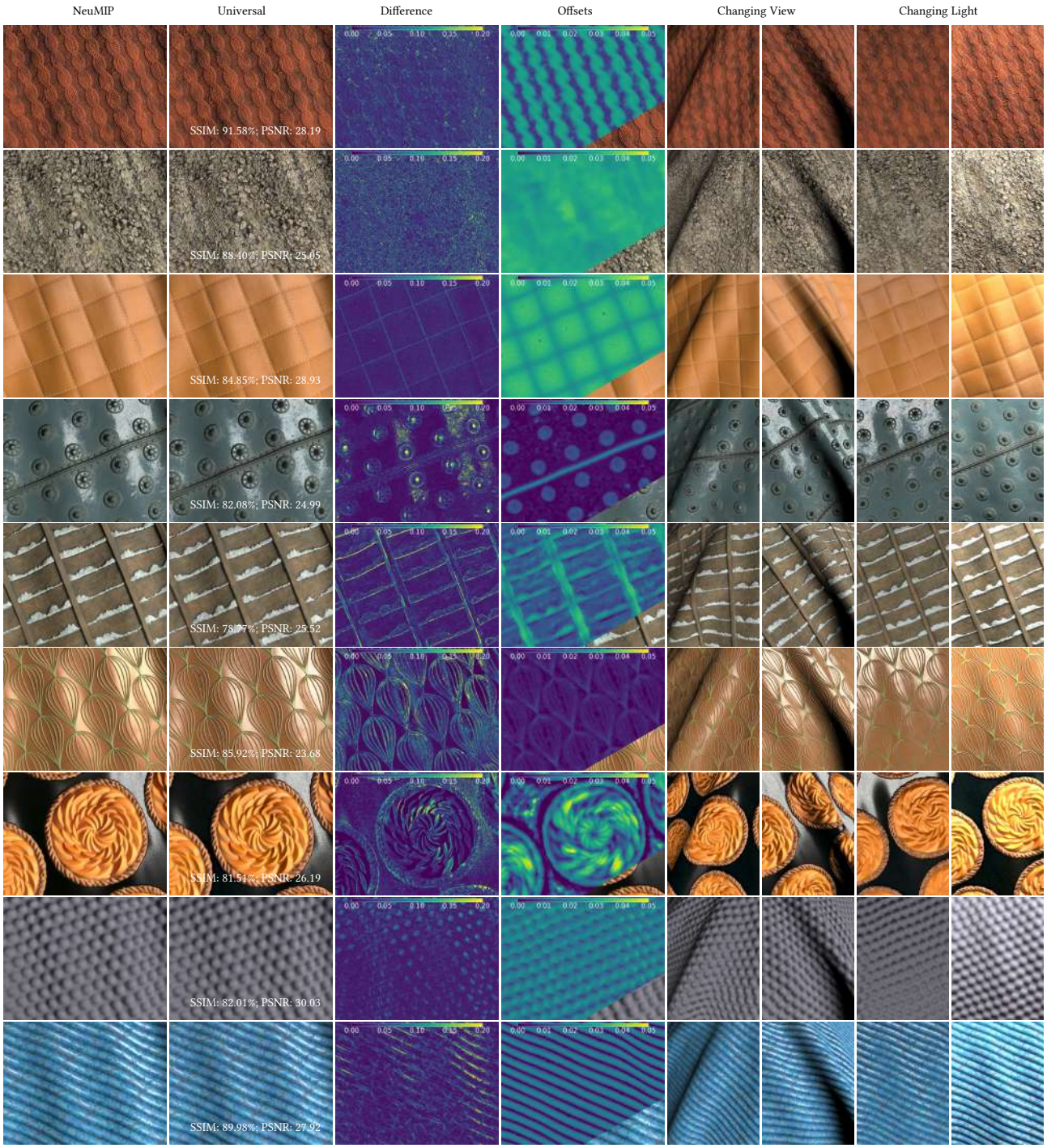

Fig. 4. **Comparison with NeuMIP.** Examples of universal materials from various categories compared against a ground-truth NeuMIP model trained individually on each dataset. The material categories, from top to bottom, are fabric, ground, leather, metal, plaster, paper, organic, plastic/rubber, and marble/granite, and are rendered on a non-flat cloth mesh. The PSNR values for both the ground-truth NeuMIP and the universal network were computed after complete training or fitting, respectively. Despite minor color deviations in the final rendered outputs, the universal texture network effectively captures diverse diffuse and glossy material properties for a variety of microgeometries and correctly determines offsets for previously unseen materials. We include more materials from additional categories in the supplementary (Figure S5).

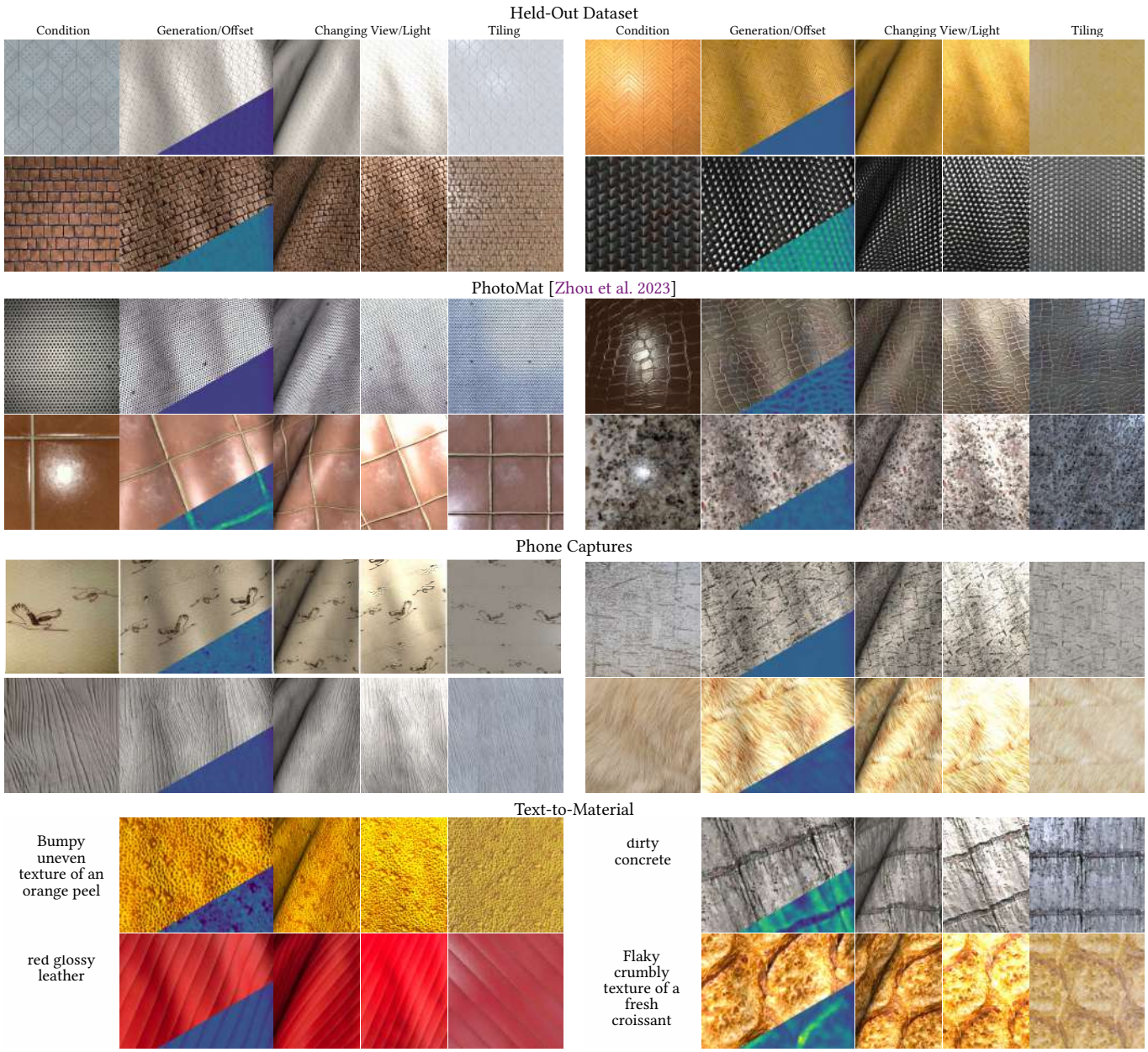

Fig. 5. **Conditional diffusion results.** We show a number of generated neural materials, conditioned on natural lighting photos, flash photos, and text prompts via a text-to-image prior ([Aggarwal et al. 2023]). For each generated material, we show multiple views and lightings on a curved surface. The offset computed by the first stage of our architecture is also visualized as an inset in the second image of each group. The rightmost image of each group is a visualization of 2×2 tiling of the resulting material, showing that the result does not show sharp cuts even if the input condition image is not tileable. We include more materials from additional categories in the supplementary (Figs. S6, S7, S8, S9, S10), and we also include a comparison with a prior SVBRDF-based generative model (MatFuse [Vecchio et al. 2024b]) in Fig. S11.

Table 1. We validate the choice of network size by comparing models trained for 30 (top section) and 100 (bottom section) epochs on a validation set of 256 materials. Our primary objective is minimizing the network size, as the network must generate conditions for our diffusion model at each iteration and integrate quickly into standard renderers. We select the smallest network that achieves the highest PSNR. As can be seen in the bottom section, training on more materials does not necessarily enhance PSNR. Thus, despite the potential for larger networks to achieve higher PSNRs with extended training, we select a depth of 4 for both M_{rgb} and M_{off} networks, trained for 100 epochs. For this table, we merged the categories of wood outdoor and wood into one, as there were not enough individually. All networks have a width of 64.

M_{rgb} depth	M_{off} depth	Num. of train mats.	Overall Average	Ceramic	Concrete/ asphalt	Fabric	Ground	Leather	Marble/ granite	Metal	Organic	Paint	Paper	Plaster	Plastic/ rubber	Stone	Terracotta	Wood
4	2	512	29.34	33.61	31.72	29.09	28.02	25.42	29.06	25.91	29.81	25.28	28.22	31.69	29.07	30.07	31.00	34.52
4	4	512	29.48	**34.15**	31.82	29.36	27.91	25.62	**29.33**	26.20	30.00	25.42	28.41	31.61	29.09	29.98	30.99	34.49
6	4	512	29.21	33.32	31.62	29.10	27.91	25.19	28.90	25.63	29.68	25.29	28.19	**31.74**	28.89	29.89	30.79	34.52
8	4	512	28.87	32.43	31.69	29.12	28.04	24.72	28.96	25.60	28.79	25.16	27.28	31.52	28.75	29.29	30.89	33.92
6	6	512	29.32	33.42	31.85	29.31	28.03	25.25	28.86	26.13	29.92	25.14	28.19	31.61	28.97	30.08	30.99	34.44
4	4	512	**29.58**	34.08	31.81	**29.54**	**28.11**	25.54	29.22	26.29	30.41	**25.58**	28.43	31.43	**29.24**	30.15	**31.08**	34.71
4	4	1015	**29.58**	33.86	**31.87**	29.38	28.05	**25.66**	29.22	**26.37**	**30.50**	**25.58**	28.49	31.46	29.20	**30.21**	31.05	**34.81**

5.2 Comparison with NeuMIP

To validate the quality of the reconstruction, we compare several fitted materials against per-material MLPs trained using the same procedure as NeuMIP [Kuznetsov et al. 2021]. For a fair comparison, the same network architecture is used for both single-material and universal models. The reference NeuMIP is trained on the same dataset used for fitting the universal MLP for 30,000 iterations, and the resulting comparisons are presented in Figure 4. The results demonstrate that our method successfully captures a diverse range of microgeometries across the material categories, as well as varied glossy and diffuse material properties. For each material, we also report PSNR values comparing the ground truth dataset with both NeuMIP and our proposed architecture.

5.3 Choice of diffusion architecture

Our architecture combines pixel-wise concatenation with cross-attention on features obtained from a CLIP ViT L/14 head. To demonstrate that such simultaneous conditioning is necessary, we provide FID results for an ablation of the CLIP module in Table S2. We evaluated this metric on 25,000 unconditional generation renders against 25,000 ground-truth dataset renders, comprising 1250 materials rendered from 20 randomly sampled viewing and environment-lighting conditions. Both methods were trained for an equal amount of time. It can be seen that the additional cross-attention module is necessary for improving the quality and diversity of samples.

5.4 Qualitative conditional generation results

In Fig. 5, we demonstrate the capabilities of our diffusion model on a number of generated neural materials, conditioned on natural lighting photos, flash photos, and text prompts. For each generated material, we show multiple views and illumination conditions, applying them to a curved surface. The offset computed by the first stage of our architecture is also visualized as an inset in the second image of each group. The rightmost image of each group is a visualization of a 2×2 tiling of the resulting material. While input conditions closely approximating tileability yield optimal results, our model can still produce tileable neural materials even when the input is not strictly tileable. However, significant deviations from tileability in the input image may lead to non-tileable outputs (see Fig. S12 in the supplementary). Notably, the model can achieve tileability along one dimension even if it fails to do so in the other. We picked our results from a variety of CFG conditions, which are illustrated in Figure S3 in the supplementary. We also observe color shifting in our diffusion model; we address the cause of this later in this section as well.

5.5 Performance

Universal basis. We benchmark the rendering performance of our universal basis network in a Vulkan-based GPU path tracing engine in Table S1. The neural material is compiled directly into the closest-hit shader. The hardware used for this benchmark is an NVIDIA RTX 6000 Ada Generation running on Windows 11. We measure the time taken to render one sample in terms of frame time by rendering a plane that has one neural material applied and fully covers the screen to 1024 samples and averaging the result. At a resolution of 1024 x 1024, we see a frame time of 37ms (27 FPS), showcasing real-time performance. Another benefit when rendering a scene with many neural materials is that unlike NeuMIP [Kuznetsov et al. 2021], our neural materials share a single MLP, enabling performance to be independent of material quantity.

Diffusion sampling. Our results were generated using 100 steps of DDIM sampling, which took 22 seconds for a 512×512 image using PyTorch.

5.6 Limitations

We observe minor color deviations in the universal network as well as more significant color deviations in the diffusion model. The latter arises from conditioning the diffusion model on diverse environment maps and point lights, which, while effective for removing highlight-related input components, introduce ambiguity in reconstructing colors under unknown illumination. Adjustments to the CFG weight during sampling (as shown in Fig. S3) mitigate this issue, though we leave it for future work to explore alternative sampling strategies for improved consistency.

6 Conclusion

We have proposed the first diffusion-based model for neural material generation. To overcome the scarcity of neural material data, we first generated a new dataset of 150k neural materials in 16 families. Given this data, we first showed how to learn a univer-

sal neural material basis; this universal basis performs comparably to the original per-material architecture while remaining broadly applicable across diverse material types. We then showed how to perform image-conditioned diffusion over these neural materials. In summary, we demonstrated a first generative diffusion model sampling neural materials from real-world image and text prompts.

Acknowledgments

We thank Liwen Wu, Bing Xu, Kai Zhang, Xilong Zhou, Lingqi Yan, Poorva Bedmutha and Alexandr Kuznetsov for comments and discussions. This work was funded in part by NSF grant IIS-2212085, and gifts from Adobe. We also acknowledge support from the Ronald L. Graham Chair, gifts from Google, Qualcomm and Rembrand, and the UC San Diego Center for Visual Computing. Ramamoorthi acknowledges a part-time appointment at NVIDIA.

References

Adobe. 2023. Substance. https://substance3d.adobe.com/assets.

Pranav Aggarwal, Hareesh Ravi, Naveen Marri, Sachin Kelkar, Fengbin Chen, Vinh Khuc, Midhun Harikumar, Ritiz Tambi, Sudharshan Reddy Kakumanu, Purvak Lapsiya, Alvin Ghouas, Sarah Saber, Malavika Ramprasad, Baldo Faieta, and Ajinkya Kale. 2023. Controlled and Conditional Text to Image Generation with Diffusion Prior. arXiv:2302.11710 [cs.CV] https://arxiv.org/abs/2302.11710

Piotr Bojanowski, Armand Joulin, David Lopez-Paz, and Arthur Szlam. 2019. Optimizing the Latent Space of Generative Networks. arXiv:1707.05776 [stat.ML] https://arxiv.org/abs/1707.05776

Katherine Crowson, Stefan Andreas Baumann, Alex Birch, Tanishq Mathew Abraham, Daniel Z Kaplan, and Enrico Shippole. 2024. Scalable High-Resolution Pixel-Space Image Synthesis with Hourglass Diffusion Transformers. In *Proceedings of the 41st International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 235)*, Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (Eds.). 9550–9575.

Kristin J. Dana, Bram van Ginneken, Shree K. Nayar, and Jan J. Koenderink. 1999. Reflectance and Texture of Real-World Surfaces. *ACM Trans. Graph.* 18, 1 (jan 1999), 1–34. doi:10.1145/300776.300778

Stavros Diolatzis, Jan Novák, Fabrice Rousselle, Jonathan Granskog, Miika Aittala, Ravi Ramamoorthi, and George Drettakis. 2023. MesoGAN: Generative Neural Reflectance Shells. *Computer Graphics Forum* 42, 6 (Sep 2023). doi:10.1111/cgf.14846

Jiahui Fan, Beibei Wang, Milos Hasan, Jian Yang, and Ling-Qi Yan. 2023. Neural Biplane Representation for BTF Rendering and Acquisition. In *ACM SIGGRAPH 2023 Conference Proceedings* (Los Angeles, CA, USA) *(SIGGRAPH '23)*. Article 81, 11 pages.

Jiahui Fan, Beibei Wang, Miloš Hašan, Jian Yang, and Ling-Qi Yan. 2022. Neural Layered BRDFs. In *Proceedings of SIGGRAPH 2022*.

Gian Favero. 2024. Simple diffusion: An annotated notebook. https://medium.com/@gian.favero/simple-diffusion-an-annotated-notebook-21ee342a6f52.

Ziyang Fu, Yash Belhe, Haolin Lu, Liwen Wu, Bing Xu, and Tzu-Mao Li. 2024. BSDF importance sampling using a diffusion model. In *Proceedings of SIGGRAPH Asia*.

Fazilet Gokbudak, Alejandro Sztrajman, Chenliang Zhou, Fangcheng Zhong, Rafal Mantiuk, and Cengiz Oztireli. 2024. Hypernetworks for Generalizable BRDF Representation. arXiv:2311.15783 [cs.GR] https://arxiv.org/abs/2311.15783

Yu Guo, Cameron Smith, Miloš Hašan, Kalyan Sunkavalli, and Shuang Zhao. 2020. MaterialGAN: Reflectance Capture using a Generative SVBRDF Model. *ACM Trans. Graph.* 39, 6 (2020), 254:1–254:13.

Tiankai Hang, Shuyang Gu, Chen Li, Jianmin Bao, Dong Chen, Han Hu, Xin Geng, and Baining Guo. 2024. Efficient Diffusion Training via Min-SNR Weighting Strategy. arXiv:2303.09556 [cs.CV] https://arxiv.org/abs/2303.09556

Zhen He, Jie Guo, Yan Zhang, Qinghao Tu, Mufan Chen, Yanwen Guo, Pengyu Wang, and Wei Dai. 2023. Text2Mat: Generating Materials from Text. (2023).

Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. 2023. Simple diffusion: End-to-end diffusion for high resolution images. arXiv:2301.11093 [cs.CV] https://arxiv.org/abs/2301.11093

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. 2019. Analyzing and Improving the Image Quality of StyleGAN. arXiv:1912.04958

Alexandr Kuznetsov, Krishna Mullia, Zexiang Xu, Miloš Hašan, and Ravi Ramamoorthi. 2021. NeuMIP: Multi-Resolution Neural Materials. *Transactions on Graphics (Proceedings of SIGGRAPH)* 40, 4, Article 175 (July 2021), 13 pages.

Alexandr Kuznetsov, Xuezheng Wang, Krishna Mullia, Fujun Luan, Zexiang Xu, Miloš Hašan, and Ravi Ramamoorthi. 2022. Rendering Neural Materials on Curved Surfaces. In *ACM SIGGRAPH 2022 Conference Proceedings*. 1–9.

Xiaohe Ma, Valentin Deschaintre, Miloš Hašan, Fujun Luan, Kun Zhou, Hongzhi Wu, and Yiwei Hu. 2024. MaterialPicker: Multi-Modal Material Generation with Diffusion Transformers. arXiv:2412.03225 [cs.CV] https://arxiv.org/abs/2412.03225

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. 2019. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. arXiv:1901.05103 [cs.CV] https://arxiv.org/abs/1901.05103

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. 2023. SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis. arXiv:2307.01952 [cs.CV] https://arxiv.org/abs/2307.01952

PolyHaven. 2025. PolyHaven: Free Public Domain 3D Assets. https://polyhaven.com. Accessed: 2025-04-25.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*. PMLR, 8748–8763.

Gilles Rainer, Adrien Bousseau, Tobias Ritschel, and George Drettakis. 2022. Neural Precomputed Radiance Transfer. *Computer Graphics Forum (Proceedings of the Eurographics conference)* 41, 2 (April 2022), 365–378. http://www-sop.inria.fr/reves/Basilic/2022/RBRD22

Gilles Rainer, Abhijeet Ghosh, Wenzel Jakob, and Tim Weyrich. 2020. Unified Neural Encoding of BTFs. *Computer Graphics Forum (Proceedings of Eurographics)* 39, 2 (2020), 167–178.

Gilles Rainer, Wenzel Jakob, Abhijeet Ghosh, and Tim Weyrich. 2019. Neural BTF Compression and Interpolation. *Computer Graphics Forum (Proceedings of Eurographics)* 38, 2 (March 2019), 235–244.

Carlos Rodriguez-Pardo, Konstantinos Kazatzis, Jorge Lopez-Moreno, and Elena Garces. 2023. NeuBTF: Neural fields for BTF encoding and transfer. *Computers & Graphics* 114 (2023), 239–246. doi:10.1016/j.cag.2023.06.018

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 10684–10695.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. arXiv:1505.04597 [cs.CV] https://arxiv.org/abs/1505.04597

Sam Sartor and Pieter Peers. 2023. Matfusion: a generative diffusion model for svbrdf capture. In *SIGGRAPH Asia 2023 Conference Papers*. 1–10.

Shufeng Tan and Michael L. Mayrovouniotis. 1995. Reducing data dimensionality through optimizing neural network inputs. *AIChE Journal* 41, 6 (1995), 1471–1480. doi:10.1002/aic.690410612 arXiv:https://aiche.onlinelibrary.wiley.com/doi/pdf/10.1002/aic.690410612

Giuseppe Vecchio, Rosalie Martin, Arthur Roullier, Adrien Kaiser, Romain Rouffet, Valentin Deschaintre, and Tamy Boubekeur. 2024a. ControlMat: A Controlled Generative Approach to Material Capture. *ACM Trans. Graph.* 43, 5, Article 164 (sep 2024), 17 pages. doi:10.1145/3688830

Giuseppe Vecchio, Renato Sortino, Simone Palazzo, and Concetto Spampinato. 2024b. Matfuse: controllable material generation with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4429–4438.

Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, Dhruv Nair, Sayak Paul, William Berman, Yiyi Xu, Steven Liu, and Thomas Wolf. 2022. Diffusers: State-of-the-art diffusion models. https://github.com/huggingface/diffusers.

Bruce Walter, Stephen R Marschner, Hongsong Li, and Kenneth E Torrance. 2007. Microfacet models for refraction through rough surfaces. In *Eurographics Symposium on Rendering*.

Bing Xu, Liwen Wu, Milos Hasan, Fujun Luan, Iliyan Georgiev, Zexiang Xu, and Ravi Ramamoorthi. 2023. NeuSample: Importance Sampling for Neural Materials. In *ACM SIGGRAPH 2023 Conference Proceedings (SIGGRAPH '23)*. Article 41, 10 pages.

Bowen Xue, Claudio Guarnera, Shuang Zhao, and Zahra Montazeri. 2024. Reflectance-Fusion: Diffusion-based text to SVBRDF Generation. In *Eurographics Symposium on Rendering*. Eurographics Association.

Tizian Zeltner*, Fabrice Rousselle*, Andrea Weidlich*, Petrik Clarberg*, Jan Novák*, Benedikt Bitterli*, Alex Evans, Tomáš Davidovič, Simon Kallweit, and Aaron Lefohn. 2024. Real-time Neural Appearance Models. *ACM Trans. Graph.* 43, 3, Article 33 (jun 2024), 17 pages. doi:10.1145/3659577

Zheng Zeng, Valentin Deschaintre, Iliyan Georgiev, Yannick Hold-Geoffroy, Yiwei Hu, Fujun Luan, Ling-Qi Yan, and Miloš Hašan. 2024. RGB↔X: Image decomposition and synthesis using material- and lighting-aware diffusion models. In *ACM SIGGRAPH 2024 Conference Papers*. 1–11.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. 2023. Adding Conditional Control to Text-to-Image Diffusion Models. arXiv:2302.05543 [cs.CV] https://arxiv.org/abs/2302.05543

Chenliang Zhou, Zheyuan Hu, Alejandro Sztrajman, Yancheng Cai, Yaru Liu, and Cengiz Oztireli. 2024. NeuMaDiff: Neural Material Synthesis via Hyperdiffusion. arXiv:2411.12015 [cs.GR] https://arxiv.org/abs/2411.12015

Xilong Zhou, Milos Hasan, Valentin Deschaintre, Paul Guerrero, Yannick Hold-Geoffroy, Kalyan Sunkavalli, and Nima Khademi Kalantari. 2023. Photomat: A material generator learned from single flash photos. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–11.

Xilong Zhou, Milos Hasan, Valentin Deschaintre, Paul Guerrero, Kalyan Sunkavalli, and Nima Khademi Kalantari. 2022. TileGen: Tileable, Controllable Material Generation and Capture. In *SIGGRAPH Asia 2022 Conference Papers* (Daegu, Republic of Korea) *(SA '22)*. Association for Computing Machinery, New York, NY, USA, Article 34, 9 pages. doi:10.1145/3550469.3555403