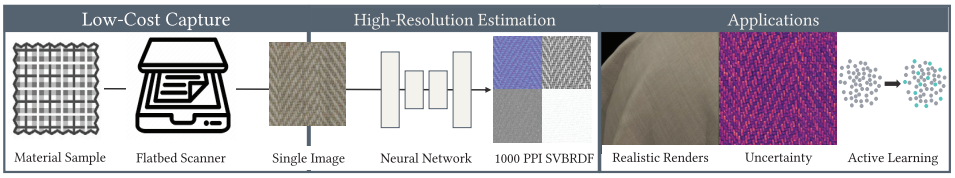
CHAPTER 5. UNCERTAINTY-AWARE NEURAL NETWORKS FOR HIGH RESOLUTION DIGITIZATION

Figure 5.1: From a single image captured with a flatbed scanner *UMat* estimates high-resolution SVBRDFs. Leveraging an attention-guided generative model, the estimations are sharp and artifact-free, and can be used in photorealistic rendering. We introduce a novel uncertainty quantification algorithm for material digitization, which correlates with error at test time and can be used for dataset creation.



In this chapter, we introduce a learning-based method to recover normals, specularity, and roughness from a single diffuse image of a material, using microgeometry appearance as our primary cue. Previous methods that work on single images tend to produce over-smooth outputs with artifacts, operate at limited resolution, or train one model per class with little room for generalization. In contrast, in this work, we propose a novel capture approach that leverages a generative network with attention and a U-Net discriminator, which shows outstanding performance integrating global information at reduced computational complexity. We showcase the performance of our method with a real dataset of digitized textile materials and show that a commodity flatbed scanner can produce the type of diffuse illumination required as input to our method. Additionally, because the problem might be ill-posed śmore than a single diffuse image might be needed to disambiguate the specular reflectionś or because the training dataset is not representative enough of the real distribution, we propose a novel framework to quantify the model's confidence about its predictions at test time. Our method is the first one to deal with the problem of modeling uncertainty in material digitization, increasing the trustworthiness of the process and enabling more intelligent strategies for dataset creation, as we demonstrate with an active learning experiment. A version of the model presented in this chapter is part of Textura.ai. The contributions in this chapter led to the following publication[1]:

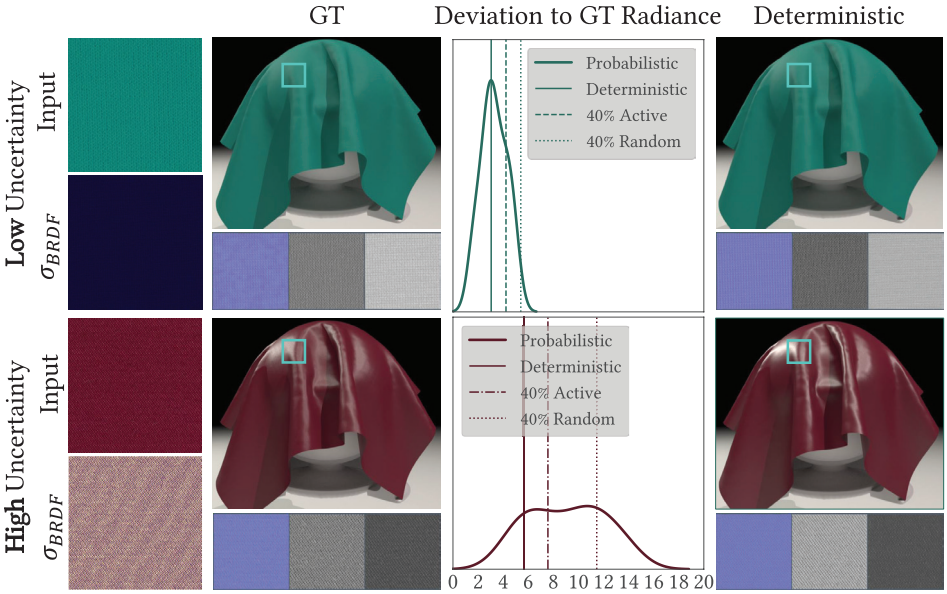
[1] Additional publication details and results are included on the project website

"UMat: Uncertainty-Aware Single Image High Resolution Material Capture"

Carlos Rodriguez-Pardo, Henar Dominguez, David Pascual, Elena Garces

Proc. of Computer Vision and Pattern Recognition (CVPR) (2023)

Figure 5.2: Our method digitizes a material taking as input a single scanned image. Further, it returns a pixel-wise metric of uncertainty σ_{BRDF}, computed at test time through probabilistic sampling, proven useful for active learning. In the plot we compare the average deviations of the radiance of different renders in the blue crop w.r.t the ground truth (GT) of: 1) the distribution of the probabilistic samples of a model trained with 100% of the data; 2) the deterministic output of that model; 3) the output of a model trained using 40% of the training dataset, sampled by active learning guided by σ_{BRDF} and; 4) a model trained using 40% of the training dataset, randomly sampled. The material at the bottom, for which the model shows a higher uncertainty, generates more varied renders and differs most from the ground truth.



5.1 Introduction

Virtual design, online marketplaces, product lifecycle workflows, AR/VR, video-games, . . . , all require lifelike digital representations of real-world materials (*i.e.*, digital twins). Acquiring these digital copies is typically a cumbersome and slow process that requires expensive machines and several manual steps, creating roadblocks for scalability, repeatability, and consistency. Among the many industries requiring digital twins of materials, the fashion industry is in a critical position; facing the demand to digitize hundreds of samples of textiles in short periods, which cannot be achieved with the current technology.

In this context, casual capture systems for optical digitization provide a promising path for scalability. These systems leverage handheld devices (such as smartphones), one or more different illuminations, and learning-based priors to estimate the material's diffuse and specular reflection lobes. However, existing approaches present several drawbacks that make them unsuitable for practical digitization workflows. Generative solutions [Guo+20b; VPS21] typically produce artifacts and are generally not tileable. Despite recent attempts to improve tileability and controllability [Zho+22], these solutions are slow to train and evaluate (requiring online optimization iterations), are limited in resolution, and present challenges for generalization (requiring one model per material class). Further, the fact that these methods build on perceptual losses śnot pixel lossesś to compare the input photo with the generated material entails extra difficulties when it comes to guaranteeing the repeatability and consistency required for building a digital inventory (i.e., color swatches, prints, or other variations). On the other hand, methods that build on differentiable node graphs [Hen+21] overcome the tileability and resolution limitations, yet, they share the problems derived from using perceptual losses and category-specific training.

In this chapter, we present *UMat*, a practical, scalable, and reliable approach to digitizing the optical appearance of textile material samples using SVBRDFs. Commonly used SVBRDFs typically contain two reflection terms: a diffuse term, parameterized by an albedo image, and a specular one, parameterized by normals, specularity, and roughness. Prior work typically estimates both

Figure 5.3: Scanner images vs fitted albedos.

components, which becomes a very challenging problem, obtaining over-smooth outputs, and being prone to artifacts [Des+18; Guo+21; ZK21]. Instead, in this chapter, we demonstrate that it is possible to provide accurate digitizations of materials leveraging as input a single diffuse image that acts as albedo and estimating the specular components using a neural network.

Our key observation is to realize that most of the appearance variability of textile materials is due to its microgeometry and that a commodity flatbed scanner can approximate the type of diffuse illumination that we require for the majority of textile materials (see Figure 5.3).

Nevertheless, single-image material estimation is still an ill-posed problem in our setting, as reflectance properties may not be directly observable from a single diffuse image. To account for these non-directly observable properties, we propose a novel way to measure the model's confidence about its prediction at test time. Leveraging Monte Carlo (MC) Dropout [GG16], we propose an uncertainty metric computed as the variance of sampling and evaluating multiple estimations for a single input in a render space. We show that this confidence directly correlates with the accuracy of the digitization, which helps identify ambiguous inputs, out-of-distribution samples, or under-represented classes. Besides increasing the trustworthiness of the capture process, our confidence quantification enables smarter strategies for dataset creation, as we demonstrate with an active learning experiment.

We pose the estimation as an Image-to-Image Translation problem (I2IT) that directly regresses roughness, specular, and normals, from a single input image. Under the hood, our novel residual architecture has a single encoder enhanced with lightweight attention modules [Wan+20b; MR21] for improving global consistency and reducing artifacts, specialized decoders for each target reflectance map, and a U-Net discriminator [SSK20], which enhances generalization.

In summary, we present the following contributions:

- A novel material capture system which leverages the diffuse illumination provided by flatbed scanners for high-resolution, scalable, and reliable digitizations.
- An attention-enhanced GAN model and training procedure designed for maximizing accuracy and sharpness, and removing undesired artifacts.
- A generic uncertainty quantification framework for material capture algorithms which correlates with prediction error on a render space.
- An exhaustive evaluation method for measuring the accuracy and quality of the model estimations.

5.2 Related Work

5.2.1 Lightweight Material Capture

Single Image SVBRDF Capture Capturing accurate SVBRDFs from a single image is a challenging problem that requires predicting the photometric response of a material given only a sample of it. The most common approximation is to use a flash-lit front planar image captured with a smartphone. Extending neural style transfer [GEB15b] to material capture, Aittala *et al.* [AAL16] leverage pre-trained CNNs and texture priors for smartphone material acquisition. Relatedly, Henzler *et al.* [Hen+21] use style losses for training generative models for BRDF synthesis. These approaches are optimized for synthesis, which allow for seamlessly tileable outputs, and do not require supervised training. However, they work best for stochastic materials, limiting their scope.

Leveraging datasets of labeled materials, different methods have trained autoencoders for SVBRDF capture. Li *et al.* [Li+17a] reconstruct spatially-varying albedo and normals using a U-Net [RFB15], and homogeneous specular albedo and roughness using a CNN regressor. This work was extended through Self-Augmented CNNs in [Ye+18]. By leveraging Conditional Random Fields (CRFs), a material classifier and one decoder per map, Li *et al.* [LSC18] reconstruct spatially-varying albedo, roughness, and normals. Deschaintre *et al.* [Des+18] propose a modified U-Net, synthetic datasets, and a render loss. Cascaded models [Li+18; SC20]; and deep latent spaces optimized using inverse rendering [Gao+19] have also shown success for this problem. Recently, Generative Adversarial Networks (GANs) have shown improved capabilities compared to more naive losses. These require a discriminator, which can be trained on renders [Wen+22], SVBRDF maps [Guo+21; VPS21], or both [ZK21].

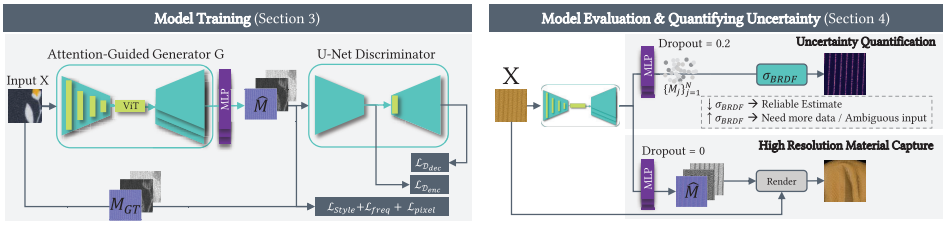
Our approach differs from previous work in several factors. Importantly, we use flatbed scanners instead of smartphones for material capture. While they limit the materials which can be captured, they provide adequate illumination for easier digitizations, and a higher level of resolution and detail. We hypothesize that material specularity can be estimated accurately by leveraging its microgeometry. From this assumption, we build a GAN which, in contrast with previous work, leverages state-of-the-art attention mechanisms and discriminator design for obtaining a more holistic understanding of its inputs. Further, we train exclusively on real data, and propose a more comprehensive evaluation.

Multiple Image SVBRDF Capture A different corpus of work relies on multiple images of the material. By capturing more evidence of the material photometric response, they provide more accurate SVBRDFs. This hinders scalability, as they require a larger capture and calibration effort. The simplest approach is to require two samples, a diffuse and a flash-lit image [AWL15; Bos+20]. More

flexible alternatives allow for more samples, combined with a learned prior [Guo+20b; Des+19]; or Monte Carlo rendering [Lua+21]. A different approach is to capture the material at a high resolution, and transfer those details to larger samples of the same material [DDB20; RG21]. Finally, videos for material acquisition have also shown to be promising capture systems [Ye+21].

Procedural Graphs Procedural graphs have also been used for material generation. Instead of relying on material priors, these approaches work by optimizing a material graph through a differentiable pipeline. These provide interesting capabilities, such as easy edition or tiling, but are limited by the expressiveness of the procedural model. These have been explored for general SVBRDF estimation [Shi+20; Hu+22c; Guo+20a], or for high-quality woven fabric digitizations [Jin+22].

Figure 5.4: Overview of *UMat*. We propose an attention-guided generator trained with style, frequency, pixel-wise, and adversarial losses. In green, we show the components that include any form of attention mechanism. The supplementary material contains the detailed architectures. On the right, we show two applications of our method: First, thanks to our test-time uncertainty quantification, we can provide a measure of the reliability of the estimation. Second, the maps that *UMat* produces can be used by any render engine.



5.2.2 Uncertainty Quantification in Deep Learning

Measuring the confidence of deep learning models is an active research area [KG17] with multiple applications, including safety-critical problems, like self-driving [Hua+19] or medicine [Kur+22]; and active dataset creation [Lei+21; Sol+21]. In computer vision, uncertainty quantification has focused on image classification [SKK18; Lei+21], segmentation [Kry+21], and depth regression [Hu+22a; Ami+20]. To overcome the computational intractability of Bayesian Neural Networks, different approximations have been proposed, including MC Dropout [GG16], Deep Ensembles [LPB17] or Variational Inference [MNG17]. Orthogonal alternatives exist, including Evidential Deep Learning [Ami+20; SKK18] or frequentist approaches like Constrained Ordinal Regression [Hu+22a]. We refer the reader to recent surveys [Jos+22; Gaw+21] for more comprehensive reviews.

Quantifying uncertainties allows communicating the end user that the model predictions may be inaccurate, suggesting alternative pathways; as well as cheaper dataset creation through *active learning*. Bayesian material parameter estimation has been proposed for procedural frameworks [Guo+20a], but, to the best of our knowledge, it has not been explored for SVBRDF estimation. We aim to propose an efficient uncertainty quantification framework for deep SVBRDF capture methods which accounts for material perception and is agnostic to the material model.

5.3 Method

Our method starts from an input image X of a material taken under diffuse lighting and outputs the parameters of the specular lobe of the SVBRDF, *i.e.*, $\mathbf{M} = \left\{\mathcal{M}_i\right\}_{i=1}^{3}$ corresponding to the material roughness, specularity, and normals. We illustrate this process in Figure 5.4.

Following previous work [KG13; Mun+22], we use the physically-based material model from Disney [BS12], which aggregates a diffuse term with an isotropic, microfacet specular GGX lobe $s(\mathbf{M})$ [Wal+07], such that, $f_{l,v}\left(\mathbf{M}, \mathrm{X}\right) = \frac{\mathrm{X}}{\pi} + S_{l,v}\left(\mathbf{M}\right) \in \mathbb{R}^{x \times y}$ is the shading model for a light l and view position v. We formulate this estimation as an Image-to-Image Translation problem (I2IT). We train a U-Net [RFB15] generator $G(\mathrm{X}) = \hat{\mathbf{M}}$ within a GAN framework, extending the adversarial loss with pixel-wise, style, and frequency losses. We design the generator to maximize accuracy, sharpness, and robustness. To do so, we train a single encoder, enhanced with self-attention layers and a transformer, and use one decoder per map. Sections 5.3.1, 5.3.2, and 5.3.3 present the network design, the loss and data augmentation choices, respectively. Implementation details are provided on the supplementary material (Section 5.A).

Using as input a single image taken under diffuse lighting presents extra challenges when estimating the SVBRDF; we lack the extra cues provided by more complex illumination patterns (e.g. flash lighting). Therefore, in Section 5.4, we explain how to compensate for this potential ambiguity by introducing an uncertainty metric that can be computed at test time. Section 5.5 presents the evaluation, which includes the description of our dataset and metrics (Sections 5.5.1 and 5.5.2), an ablation study that validates the design (5.6.1), qualitative and quantitative results (5.6.2 and 5.6.3), an application of our uncertainty metric in an active learning setting (5.6.4), and comparisons with previous work (5.6.5).

5.3.1 Network Design

We use a U-Net [RFB15] with residual connections [He+16; Dia+20; DLG21] in all of our convolutional blocks, an individual decoder per map [Gar+22; RG22;

DLG21; ZK21], and group normalization [WH18]. To each decoder, we add a pixel-wise dropout-regularized MLP [Sri+14], aimed at increasing the accuracy of the predictions and allowing us to measure uncertainty at test-time.

While this multi-decoder residual U-Net is relatively accurate, it is limited by its receptive field, as is common on fully-convolutional architectures. Previous work [Des+18] proposed the use of a *global track* for fusing spatially distant information. We instead draw inspiration from recent advances in attention and diffusion models [Sah+22; HJA20; Rom+22], and add a self-attention module with linear complexity [Wan+20b] to the output of every convolutional block in the encoder. Finally, we add a lightweight MobileViT [MR21] to the bottleneck to provide the model with a global understanding of its input. By performing the most complex computations at the encoder, we provide the specialized decoders with dense inputs which are computed only once.

5.3.2 Loss Function

Our loss function is comprised of four terms: pixel-wise losses, an adversarial loss, a style loss, and a frequency loss:

$$\mathcal{L}_G = \sum_i \lambda_i \mathcal{L}_{pixel_i} + \lambda_{adv}\mathcal{L}_{adv} + \lambda_{style}\mathcal{L}_{style} + \lambda_{freq}\mathcal{L}_{freq} \tag{5.1}$$

L_{pixel} is the L_1 norm weighted per map, λ_i. L_1 produces sharper results than higher-order alternatives, such as L_2. We introduce an adversarial loss to handle the intrinsic ambiguity of ill-posed problems [Tex+20b; Iso+17; Gar+22; VPS21]. In our case, the choice of the discriminator is a critical design decision.

Recent work [SSK20] proposed U-Net architectures for discriminators, which allows for better learn both low and high-level features, and introduces further regularization. These result in more conservative albeit less diverse generations [Han+22a]. We use a U-Net discriminator [SSK20] with attention [Woo+18], which outputs two estimations: a scalar output D_{enc}, provided by its encoder, and a 2D estimation D_{dec}, provided by its decoder. D_{enc} provides a global estimate of the quality of the stack M, while D_{dec} gives pi estimations. As in [SSK20], we add a *regularization* term $\mathcal{L}^{cons}_{D_{dec}}$, and leverage *cut-mix* as for discriminator data-augmentation. Our discriminator and adversarial losses are:

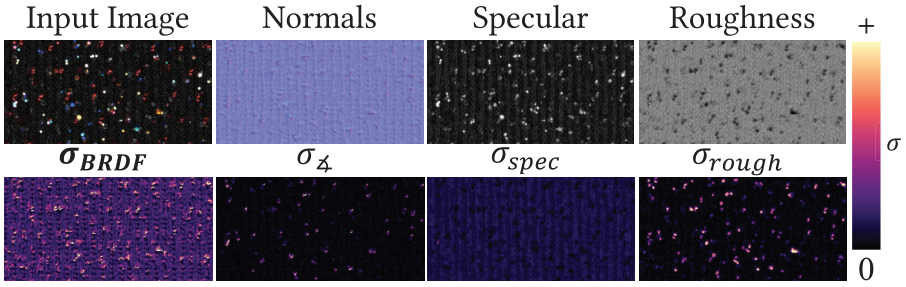
$$\mathcal{L}_D = \mathcal{L}_{D_{enc}} + \mathcal{L}_{D_{dec}} + \lambda_{cons}\mathcal{L}^{cons}_{D_{dec}} \tag{5.2}$$

$$\mathcal{L}_{adv} = \log\left(\mathcal{D}_{enc}\left(\mathrm{G}\left(\mathrm{X}\right)\right)\right) + \log\left(\mathcal{D}_{dec}\left(\mathrm{G}\left(\mathrm{X}\right)\right)\right) \tag{5.2}$$

where the implementation of $L_{D_{enc}}$ and $L_{D_{dec}}$ follows [SSK20].

We further add two losses to improve the accuracy and sharpness of the results: a frequency loss L_{freq} and a style loss L_{style}. L_{freq} is estimated by averaging the *focal frequency loss* [Jia+21] computed over each individual channel of M. This is designed to help GANs preserve high-frequency details. Further, as shown in prior work, working in the frequency domain is beneficial when handling textures [AWL13; Mar+20]. Our style loss L_{style} is inspired by the success of neural losses when dealing with textures [GEB15b; GEB15a]. However, off-the-shelf metrics which are designed for 3-channel images, are not immediately usable in SVBRDF. While it is possible to compute them for each map separetely [RG22], this does not necessarily preserve inter-map consistency. We follow recent work [CHB21] and use LPIPS [Zha+18] taking as input a 3-channel image created by randomly sampling three channels from the set of five available channels of M.

Figure 5.5: Top: input image of a *rib* material with metallic sequins. Bottom: σ_{BRDF} and per-map uncertainties.



5.3.3 Data Augmentation

We follow two strategies for data augmentation. First, we perform patch-based training and affine transforms with randomly cropped patches [RG21; Tex+20b; VPS21]. We also apply random rescales for generalization at lower resolutions, and rotations to account for possible misalignments that may arise when capturing the samples. Second, we apply several image transformations to increase model robustness: random intensity changes in HSV space to the inputs, gaussian noise and blurs, and random erasing [Zho+20] for regularization.

5.4 Uncertainty Quantification

Material acquisition from a single diffuse image is potentially an ill-posed problem, since it assumes that the microgeometry is a sufficient cue to predict the material appearance. While purely learned priors have been proven to work well for inverse problems of single images object shape estimation [Lic+21; Lua+21; Hwa+22; Mun+22; Li+18], they are typically combined

with render losses that guarantee consistency in the reconstruction. We lack the necessary input to include this kind of supervision, therefore, we propose an alternative approach aimed at quantifying the confidence of the prediction.

This is valuable for several purposes. First, it provides a way of communicating possible inaccuracies to users of these systems when it is not possible to have access to the ground truth reflectance values; and, importantly, it enables efficient dataset creation through active learning, as we show in Section 5.6.4.

We propose an uncertainty quantification mechanism that is applied to individual per-map estimations, and also globally in a render space.

It is possible to measure uncertainty, among other methods, through deep ensembles [LPB17], evidential learning [SKK18; BYK21; Wan+22b], or ordinal regression [Hu+22a]. However, they are costly to train and evaluate, and would imply major changes in our method. Instead, we follow a probabilistic approach called MC Dropout [GG16], with which, for a particular input, we sample a set of predictions by adding randomness to the forward pass of our model. This process has no impact in the regular deterministic evaluation and implies no changes in our model architecture. Specifically, while measuring uncertainty, we randomly deactivate 20% of the neurons of the MLP of our decoders, obtaining a set of outputs $U = \left\{ \hat{\mathcal{M}}_j \sim \mathrm{G}(\mathrm{X}) \right\}_{j=1}^{N}$ that we use to compute several metrics.

First, we compute the pixel-wise standard deviation for each invididual map separately, obtaining $\sigma_{\vec{n}}$,σ_{spec}, and σ_{rough}, for the normals, specular, and roughness maps, respectively. Then, inspired by perceptual metrics for computing BRDF differences [Lav+21], we define our novel perceptually-aware uncertainty metric σ_{BRDF} as follows:

$$\sigma_{\text{BRDF}} = \frac{1}{|xy|} \sum_{xy} \log\left(\frac{1}{|S|} \sqrt{ \sum_{(l,v)\in S} \sqrt[3]{ \sigma_{l,v}^2 \left(\left\{ f_{l,v}\left(\mathrm{U}_j, K\right) \cos\left(\theta_l\right) \right\}_{j=1}^{N} \right) } } \right) \tag{5.4}$$

where K is a 2D grayscale image of a constant neutral grey value, $\sigma_{l,v} \in \mathbb{R}^{x \times y}$ is the pixel-wise standard deviation of the renders obtained for the set of sampled maps U, and S is the fixed set of 50 views optimized in [NJR15] for efficient BRDF capture. The log spatially-varying result is integrated across the spatial dimensions xy to obtain a single value as output. This equation introduces a perceptual component to our uncertainty metric in two forms: first, by applying cosine weighting to the light position to compensate for light attenuation at grazing angles, and second, by taking the cubic root of these differences to attenuate peak reflectances. Figure 5.5 showcases an

example of the predicted uncertainties for a rib material with sequins. While the uncertainty is low at the yarns because it is a common material in our dataset, it appears high in the sequins since that effect had not been observed during training.

5.5 Experimental Results

5.5.1 Dataset

We gathered a novel dataset for training and testing our method. It comprises 2000 textile materials of a variety of families and microstructures that we divided into 14 families: crepe, jacquard, pile, plain, satin, and twill for *wovens*; fleece, french terry, interlock, jersey, milano, pique, and rib for *knits*; and, finally, *leathers*. Each family differs in its construction pattern

(*i.e.*, its microstructure), which directly impacts its optical appearance. In the supplementary material, we show a more detailed analysis of the dataset. For each material, we have an image scanned with the flatbed scanner EPSON V850 whose lighting configuration is close to diffuse (as we show in Figure 5.3), and its corresponding ground truth specular maps of the SVBRDF (normals, specular, and roughness). To obtain these maps, first, we digitized the material with an optical gonioreflectometer and then, propagated the maps to the scan using map propagation techniques [RG21]. All our images and maps have a resolution of 1000 PPIs, allowing us to leverage the full semantics of the microstructure for inference. We split our dataset into 90-10 for train and test, making sure that every family is equally represented in both splits.

5.5.2 Metrics

We quantify individual *per-map accuracy*, *rendered perceptual accuracy*, and *artifacts*. **Per-map accuracy** is computed differently depending on the semantics of the map: the roughness and specular maps are evaluated using Mean Absolute Error (L_1), and the normals are evaluated using the angular distance in vector space ($L<$). Further, to account for the possibility that the model always returns an accurate average value, resulting in relatively low L_1, we also measure Pearson correlation ρ.

We evaluate **rendered perceptual accuracy** LBRDF between the ground truth stack $\mathbf{M}_{GT}$ and the estimation $\hat{\mathbf{M}}$following existing metrics [Lav+21],

$$\mathcal{L}_{\text{BRDF}} = \frac{1}{|xy|}\sum_{xy}\sqrt{\frac{1}{|S|}\sum_{(l,v)\in S}\sqrt[3]{\cos^2\left(\theta_l\right)\left(f_{l,v}\left(\mathbf{M}_{GT},K\right)-f_{l,v}\left(\hat{\mathbf{M}},K\right)\right)^2}} \tag{5.5}$$

where the terms are the same as in Equation 5.4.

Finally, we found that some architectural improvements in the neural network introduce artifacts in the specular and roughness maps that were not present in the input image, as illustrated in Figure 5.6. Thus, we provide an **artifacts detection** metric to quantify them. We start by defining a metric of homogeneity for an input image $H(\mathrm{I})$,

$$\mathcal{H}(\mathrm{I}) = \frac{1}{|xy|}\sum_{xy}\frac{1}{|d|}\sum_{d=\{\uparrow,\downarrow,\leftarrow,\rightarrow\}}\left\| F_{\mathrm{Box}}(\mathrm{I}) - F_{\mathrm{Box}}(\mathrm{I}^d)\right\|_1 \tag{5.6}$$

where I^d is the image shifted up, down, left, and right by a number of pixels equal to the kernel size of the box filter. Then, we define three metrics that we compute per map: $e_2(\mathcal{M}) = \frac{\mathcal{H}(\mathcal{M})}{\mathcal{H}(\mathrm{X})}$, and $e_3(M) = MI(X,M)^{-1}$. X is the original input image, and *MI* is the Mutual Information [Rus+04], which helps discern artifacts that appear in a single map from semantic patterns (*e.g.,* plaids, prints). Each map is labeled as having artifacts if the majority of metrics exceed their corresponding thresholds, $t_m(M)$. If any of the maps contain artifacts, the entire stack **M** is classified as having artifacts.

5.6 Evaluation

5.6.1 Ablation Study

In Figure 5.6 and Table 5.1, we present an ablation study to validate our model design. From a baseline U-Net [RFB15] trained with a pixel-wise loss and no data augmentation other than rescales, we add different components to the model to improve its generalization. First, we observe that using a PatchGAN [Iso+17; VPS21] discriminator provides a significant increase in accuracy. However, using a similarly-sized U-Net discriminator [SSK20], we achieve better results, particularly when using discriminator regularization. Further, L_{style} and L_{freq} yield significant improvements, most notably in the normal map. To the baseline U-Net trained with the full loss, adding residual connections and one decoder per map increases accuracy. However, this setup tends to produce artifacts as it struggles to integrate global information. By adding self-attention to the encoder, we remove the artifacts; and with the MobileViT [MR21], we achieve higher quality results. Our final model contains full data augmentation, which provides small gains on generalization.

5.6.2 Qualitative Analysis

We aim to understand which features are exploited by our model for making its predictions. In Figure 5.7, we show the embeddings of our generator using UMAP. It seems that our model learns to separate between material *families*

(*e.g. leathers* from *wovens*). Interestingly, the *thickness* of the material is also a relevant parameter. The model is exploiting these semantic patterns without explicit supervision, providing evidence that material microgeometry plays important an important role in its optical appearance.

Figure 5.6: Qualitative results of some configurations of our ablation study. In **red**, we show that the baseline generator architecture trained on the full loss introduces artifacts, which are removed using **attention** on the encoder.

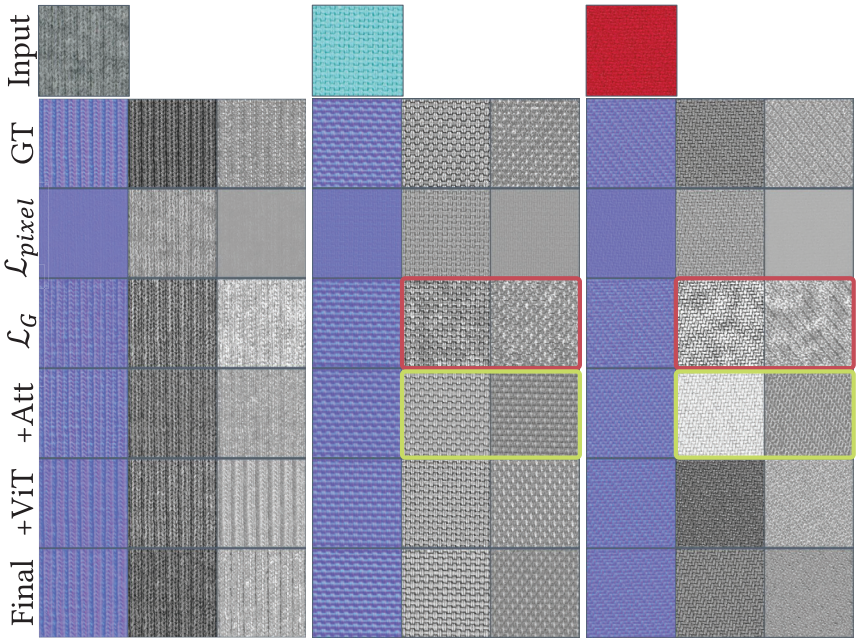


Figure 5.7: Embeddings of the transformer of our generator for data in the training set, reduced using UMAP [MHM18].

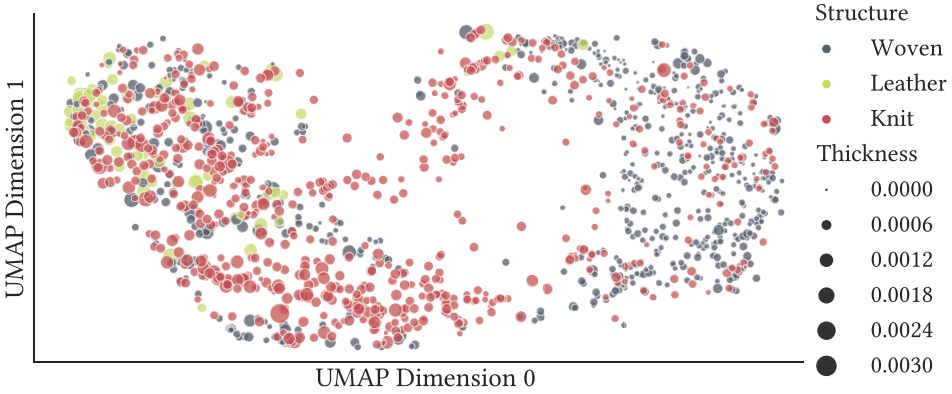


Table 5.1: Results of our ablation study, across a variety of metrics. *Art.* refers to our artifact detection metric. We use a color code to highlight **best** and **worst** cases.

	Configuration	$\rho^S\uparrow$	$\rho^R\uparrow$	$L_{\not\perp}\downarrow$	$\mathcal{L}_1^S\downarrow$	$\mathcal{L}_1^R\downarrow$	$L_{BRDF}\downarrow$	Art.↓
	Baseline	0.615	0.329	7.570	0.130	0.070	0.325	0.0
Loss	Baseline + PatchGAN	0.810	0.510	3.729	0.089	0.069	0.307	12.0
	Baseline + U-Net D.	0.854	0.658	2.950	0.085	0.068	0.299	28.0
	U-Net D. + $\mathcal{L}^{cons}_{\mathcal{D}_{dec}}$	0.858	0.653	2.860	0.088	0.068	0.296	**19.0**
	$\mathcal{L}^{cons}_{\mathcal{D}_{dec}}$ + L_{style}	0.831	0.655	2.790	0.091	0.060	0.289	**23.0**
	L_{style} + L_{freq}	0.856	0.677	2.410	0.086	0.059	0.288	**20.1**
Model	L_{freq} + Residual	0.855	0.665	2.310	0.089	0.060	0.285	**25.5**
	Residual + Decoders	0.863	0.699	2.120	0.079	0.054	0.276	**18.5**
	Decoders + Attention	0.860	0.665	2.080	0.080	0.059	0.275	**0.5**
	Atention + ViT	0.863	0.665	2.040	0.079	0.057	0.271	**0.5**
Augmentation	ViT + Color	0.899	0.692	1.969	0.068	0.054	0.269	**0.0**
	Color + Rotations	0.870	0.682	2.050	0.078	0.055	0.271	**0.2**
	Rotations + Distortion	0.876	0.699	2.001	0.074	0.053	0.268	**0.0**
	Distortion + Erasing	0.893	0.727	1.941	0.067	0.052	0.265	**0.0**

5.6.3 Uncertainty Evaluation

In Figure 5.8, we show the Pearson correlation matrix between the uncertainty and error for each map, our uncertainty metric (Equation 5.4), and the render perceptual metric (Equation 5.5), on our test set. As shown, neither the error nor uncertainties per map explain errors on the render space. Our proposed σ_{BRDF} achieves a remarkable correlation of 82% with the render error, validating that this metric is useful to predict errors at test time with reasonably high precision.

In the same plot at the bottom, we distill the uncertainty and render error per family, in which we observe that our model struggles to accurately and confidently predict the reflectance of *satins*, *jacquards*, and *leathers* more than it does for any other structure in our dataset. These structures have complex optical behavior that makes their digitization more challenging (for example, satins exhibit anisotropy, which we do not support in our material model), and are relatively uncommon in our training dataset. Figures 5.2 and 5.5 show further examples of our uncertainty estimation for a diverse set of materials.

5.6.4 Active Learning

We leverage σ_{BRDF} for active learning [Sol+21] to identify the samples that contribute most to reduce the error of the model. Figure 5.9 (top) illustrates the process. We start by training a model with 10% of the available training data and measure the uncertainties in the remainder of the dataset. We then select the samples with the highest uncertainties and re-train the model with 20%. We repeat the process with 40, 60, 80 and 100% of the training dataset. For each subset, we compare the performance of this model with four baselines: random-sampling, sampling by the highest uncertainty in normals, roughness, and specular. The results are shown in Figure 5.9 (bottom). Active sampling based on σ_{BRDF} provides significant gains in sample efficiency, obtaining better accuracy for every map. For instance, an actively trained model with our metric which uses 20% of the training data obtains comparable results to a model trained on three times more (but randomly sampled) data. While $\sigma_{\text{}}$ is typically more informative than σ_{rough} and σ_{spec}, using per-map uncertainties does not provide better results than a random strategy. Finally, Figure 5.2 shows the variation of the probabilistic samples with respect to the ground truth radiance for two materials with high and low uncertainty.

Figure 5.8: Top-left, correlation between render and pixel-wise losses, and uncertainties. Top-right, plot showing the correlation between our uncertainty metric and the error in render space; the renders illustrate the worst and best cases. Bottom, uncertainty and errors for different material families of the test set.

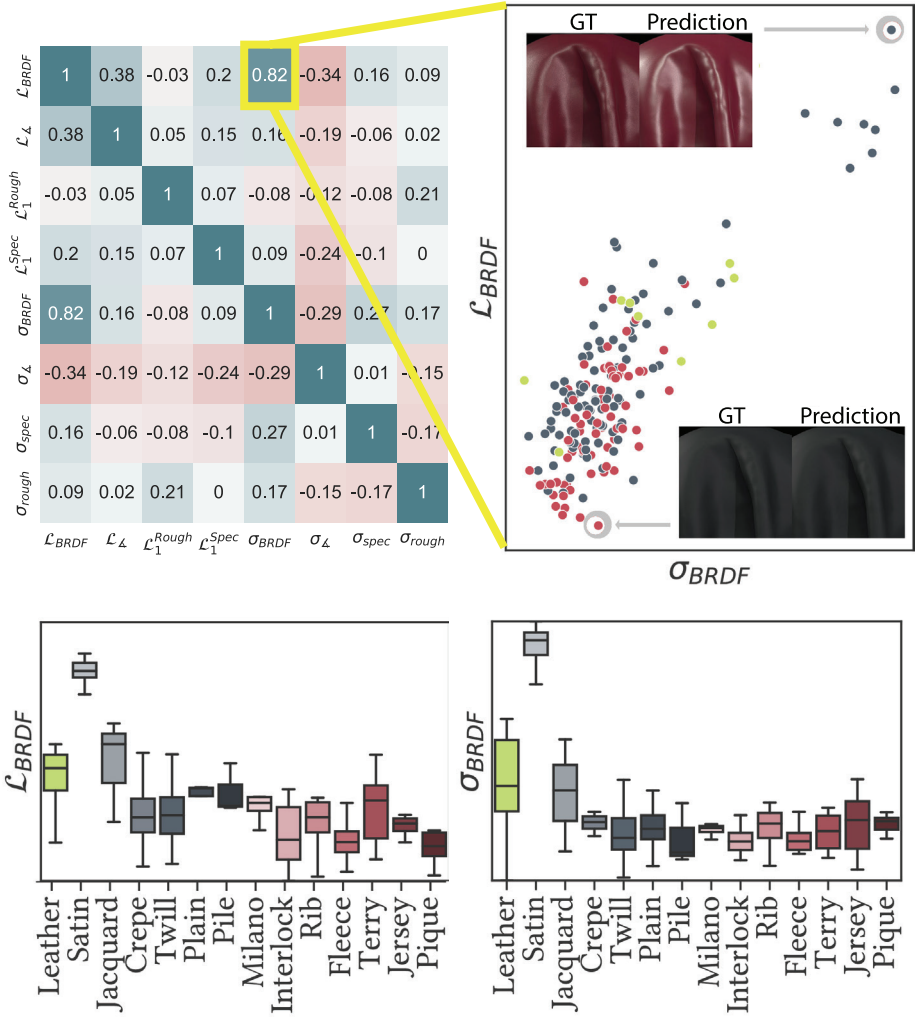


Figure 5.9: On top, illustration of our active learning algorithm. On the bottom, results of our active learning experiments. Leveraging σ_{BRDF} for actively selecting the top-k samples with the highest uncertainty, we achieve better results than a random sampling strategy for every metric we measure.

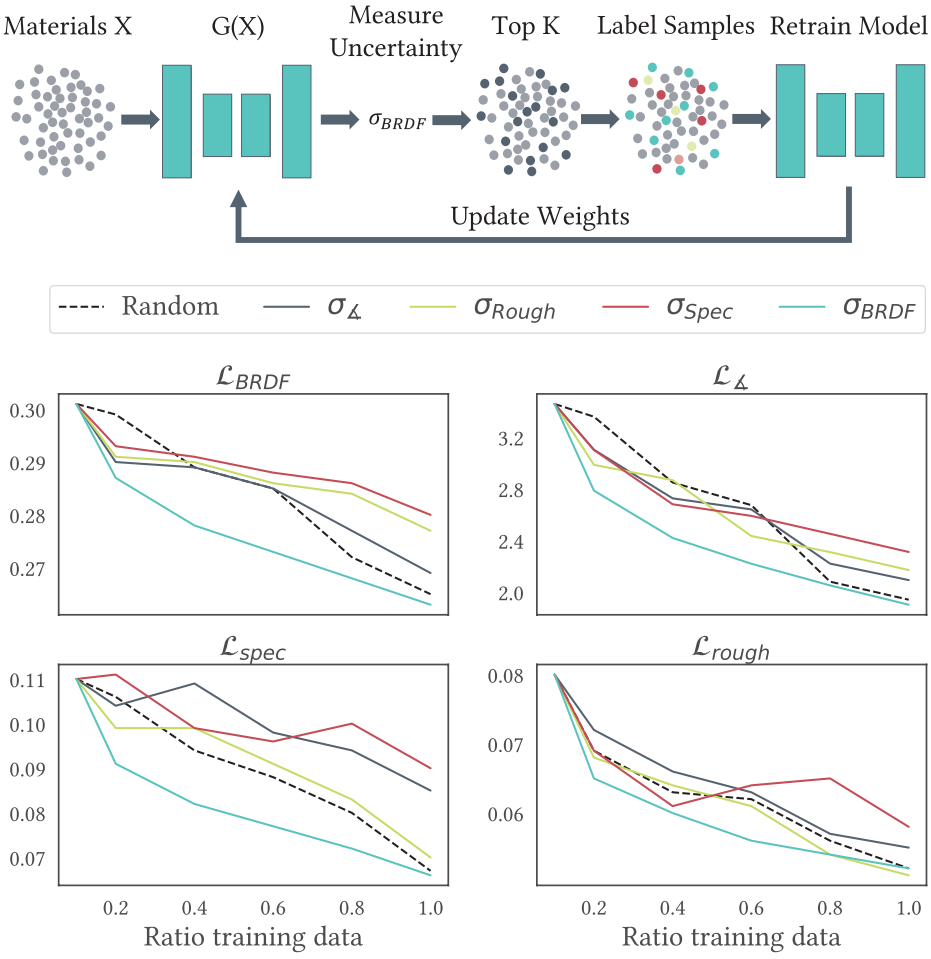


Table 5.2: Model sizes for different methods, and evaluation time in seconds (mean of 100 evaluations on an RTX 2080 GPU), for different output sizes. The methods with * use test-time optimization. DiffMat [Shi+20] does not use a pre-trained model.

Method	Size (MB)	Output Dims	Eval Time (s)
Deep Inverse Rendering* [Gao+19]	167	256×256	603.5
Generative Modeling* [Hen+21]	1095.3	512×384	218.8
Diff. Material Graphs* [Shi+20]	-	512×512	1209.8
Adversarial Estimation [ZK21]	11552.2	256×256	0.078
UMat (Ours)	22.6	256×256 512×512	**0.036 0.131**

5.6.5 Comparisons with Previous Work

In Table 5.3, we compare our method with previous work on single image material capture. First, we made sure that the training data for these methods included textile materials similar to the ones we choose for testing. Emulating their capture conditions, we took images with a smartphone, with flash and ambient lighting. Note that these capture conditions are not ideal for our method, affecting the final renders if the albedo has shading gradients. However, our goal in this experiment is to evaluate the overall preservation of the material structure in the inferred maps, particularly visible in the normals.

For Shi *et al.* [Shi+20], we initialize the graph using a fabric material, provided in their open-source implementation and include the metallic map. Our model provides sharper and more accurate estimations without requiring optimization during test. Methods trained on style losses [Hen+21; Shi+20] degrade the semantic structure, while Zhou *et al.* [ZK21] generate similar albedos to ours (note that ours are captured), but provide over-smooth estimations. We provide a comparison of timings and model sizes in Table 5.2. With our efficient model design, we can provide real-time estimations without needing any optimization, which also enables our sampling-based uncertainty quantification.

Table 5.3: Comparisons of our method with previous work on images captured under different lighting conditions. Top: a smartphone flash-lit image. Middle: a smartphone image with ambient light. Bottom: a flatbed scanner capture. Our method produces the best results preserving the microstructure even when capture conditions degrade due to the sensor resolution. Note that we do not estimate albedos and that absolute intensities for specular and roughness maps are not directly comparable due to differences in the material model.

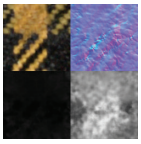
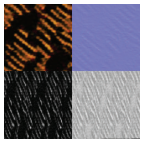
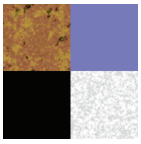
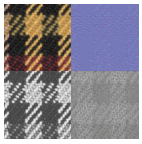
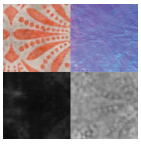
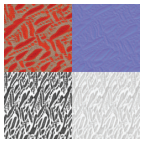
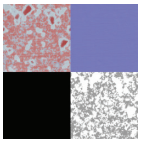
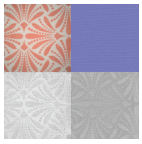
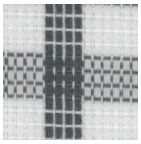
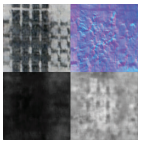
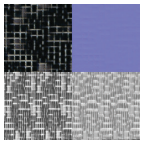
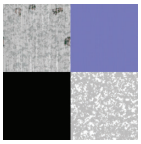
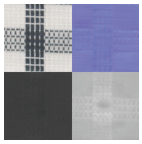
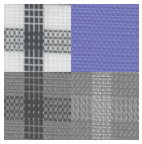
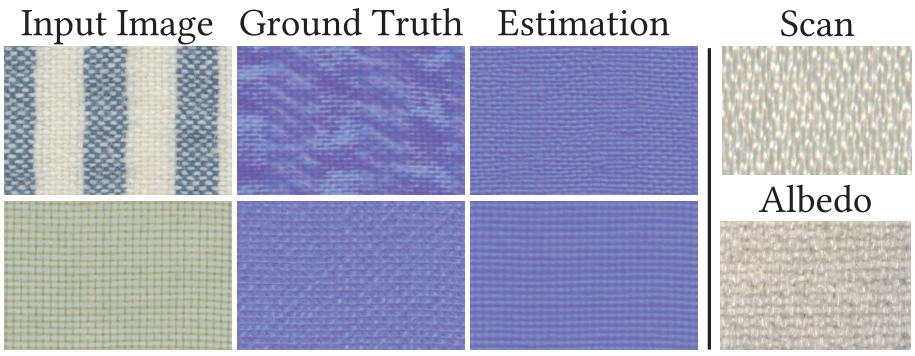
IInput	Deep Inverse R. [Gao+19]	Generative Model [Hen+21]	Diff. Material Graphs [Shi+20]	Adversarial Est. [ZK21]	**UMat (Ours)**
					
					
					

Figure 5.10: Limitations cases of our method. On the left, we show a *seersucker*, with wrinkles that are hidden by the diffuse illumination of the device, and a translucent *organza* with holes between the yarns that appear very bright due to the white background of the scanner, and are therefore mistakenly treated as yarn centers. On the right, we show that for highly directional materials, such as *satins*, the diffuse-like illumination in our capture device sometimes introduces specular highlights.



5.6.6 Limitations

We show some limitations in Figure 5.10. The illumination in the scanner hides the wrinkles in the *seersucker* fabric, and our model predicts a flat surface. The *organza* fabric at the bottom is very transparent with visible holes between the yarns. Since the background is white, the model has mistakenly treated the light regions as yarn centers. For the *satin* at the right, the scan image exhibits specular highlights due to the directionality of the yarns. While this image may be problematic to use as an albedo, it does not affect our metrics as we use constant albedos to compute them.

5.7 Conclusions

We have presented a GAN-based method to digitize materials which leverages microgeometry appearance and a flatbed scanner as a capture device. Our method has shown better performance than state-of-the-art solutions that require a single image as input, when it comes to textile materials. To account for potential ambiguities derived from the capture setting, we have presented a method to model the uncertainty in the estimation at test time.

Managing uncertainty in machine learning projects is important to guarantee robust and functional solutions. However, this typically comes at the cost of complex or slow models. In this work, we have presented the first method to quantify uncertainty in single image material digitization, while introducing minimal impact in the training and evaluation processes. While it is currently not possible to discern the source of the uncertainty, whether it is epistemic (uncertainty which can be reduced by increasing the dataset size) or aleatoric (which is derived by a noisy data generation process), our metric has proven useful to identify ambiguous inputs, underrepresented classes, or out-of-distribution data.

We could extend our work in several ways. The most obvious extension is to estimate real albedos, so that we can deal with other types of scanning devices. Further, expanding our material model to give support for more reflectance properties, such as transmittance or anisotropy, could be useful to improve the realism in the render of textiles.

5.A Additional Implementation Details

5.A.1 Model Design

Our model is trained using a GAN framework. In this section, we detail our design choices for the generator and discriminator architecture.

Generator For the generator, we use a U-Net [RFB15] model, with a few modifications designed to maximize its efficiency, robustness, and generalization capabilities. We specify the full model architecture and layer sizes in Figure 5.11. We use residual connections [He+16; Dia+20; DLG21] in every convolutional block of the model, for better training convergence and preserving details present in the input images. We use 1×1 convolutions on the skip connections. Further, to maximally preserve the appearance and characteristics of every target map, we use a single decoder for each. This has been proposed in different applications, including intrinsic images, material capture, and texture synthesis [Gar+22; RG22; DLG21; ZK21]. To improve the results and enable our uncertainty metric, we append a pixel-wise MLP to each decoder in the model, with Dropout [Sri+14] regularization. Using MLPs after the decoders has been previously explored for material capture [Des+19]. We use Group Normalization [WH18] (with 16 *groups* per layer) and SiLU [EUD18] non-linearities throughout the model. Each convolutional block in the encoder is enhanced with a lightweight Linear Attention module [Wan+20b], with 4 *attention heads*, each with a dimension of 32 hidden units. On the bottleneck, we use a lightweight *MobileViT* Transformer block [MR21], with 128 hidden dimensions for the self-attention and MLPs, 4 layers, and a kernel size of 3. We use a Dropout [GG16] rate of 0.2. We use transposed convolutions for upsampling. Every other implementation detail in the model (strides, bias, poolings) follows [RFB15].

Discriminator For the discriminator [SSK20], we also use a U-Net [RFB15] model, with a few modifications to improve its performance as a discriminator. We specify the full model architecture and layer sizes in Figure 5.12. As in the generator, we use residual connections [He+16; Dia+20; DLG21] in every convolutional block of the model, for better training convergence and preserving details present in the input images. We use Spectral Normalization [Miy+18] and SiLU [EUD18] non-linearities throughout the model. On the bottleneck, we use a lightweight *CBAM* attention block [Woo+18]. The single-scalar estimation of the discriminator D_{enc} is provided by an MLP with a similar architecture to the *CBAM* module. We initialize the entire model using orthogonal initialization [HXP20]. We use transposed convolutions for upsampling. Every other implementation detail in the model (strides, bias, pooling) follows [RFB15].

5.A.2 Model Training

Optimization We train the models using PyTorch [Pas+19] and TorchVision [MR10]. We leverage Kornia for data augmentation [Rib+20]. To accelerate the training process, we leverage mixed precision training and automatic gradient scaling [Mic+18], and train the whole model natively on GPU. Optimization is done using Adam [KB15]. Following [SSK20], we use different learning rates for the generator lr = 0.001 and the discriminator lr = 0.005 and a batch size

of 10, betas= (0.9, 0.99) and a weight decay of 0.000003. We train the models for 100 epochs, which takes 10 hours on an NVIDIA RTX 3060 GPU.

Loss Function We use the following weights for the loss function: $\lambda_{\lambda} = 3$, $\lambda_{spec} = 1, \lambda_{rough} = 1, \lambda_{adv} = 0.2, \lambda_{style} = 0.25, \lambda_{freq} = 0.2, \lambda_{cons} = 0.3$. For the style loss function, we use the *AlexNet* variant of LPIPS [Zha+18], as it provides a lightweight style loss which has shown success on texture transfer [RG21].

Data Augmentation Training is done using random crops of 128 × 128 pixels. We perform random rescales uniformly on the 300, 1000 PPI range, with bilinear interpolation. We randomly rotate the SVBRDF on the [−10, 10] angle range. We use the algorithm in [RG21] to correctly rotate the normal map. On the HSV color space, we randomly change the properties of the model inputs. We randomly change the hue of the images, selected at random on the whole hue ranges. We also change the saturation, value, and contrast of the input images with factors of [0.9, 1.1], as specified on the Torchvision [MR10] ColorJitter implementation. With a probability of 0.3, we apply random Gaussian noise to the input images μ = 0, σ = 0.10. With a probability of 0.35, we apply random erasing [Zho+20] to the input images. Half of the time, we apply a random Gaussian blur to the input images $\sigma \sim U$ (0.1, 5). Finally, with a probability of 0.3, we apply cut-mix data augmentation to the discriminator, as in [SSK20].

Model Evaluation We use half precision for model evaluation, which makes the process faster and allows us to generate SVBRDF of larger sizes. For images of very high resolutions (eg ≥ 4000 × 4000), we use the *stitching* algorithm in [DDB20], with patch sizes of 2048 and stride of 1024 pixels.

5.A.3 Artifact Detection

The thresholds for each material map and the kernel size for the uniformity metric have been optimized given a set of 102 manually labeled textures: $t_1(M_s) = 0.01$, $t_2(M_s) = 1.41$,

$t_3(M_s) = 1.33$; $t_1(M_r) = 0.01$, $t_2(M_r) = 0.99$, $t_3(M_r) = 3.12$. The size of the box filter is $s_{\text{Box}} = 0.1275 \cdot \text{DPI}_X$. The thresholds for each material map and the kernel size for the uniformity metric have been optimized given a set of 102 manually labeled textures: $t_1(M_s) = 0.01, t_2(M_s) = 1.41$, $t_3(M_s) = 1.33$; $t_1(M_r) = 0.01$, $t_2(M_r) = 0.99$, $t_3(M_r) = 3.12$. The size of the box filter is $s_{\text{Box}} = 127.5$.

5.A.4 Capture Details

We construct the training and testing dataset capturing 10 × 10 cm samples at 1000 PPI using an *EPSON V850 Pro* on its default settings.

For the comparisons with previous work, we place the fabric samples on a black surface. We use a *Huawei Nova 5T* smartphone, and capture the materials at two distances: a *close-up*, capturing 4 × 4 cms at a distance to the sample of 8 cms, and a *full-size* image, capturing 8 × 8 cms, at a distance to the sample of 13 cms. We use ISO=50, an aperture of $\frac{f}{1.8}$, and a focal length of 26mm for every image. For the two distances, we capture the material using ambient illumination, with exposure times of $\frac{1}{10}s$ for the *full size* and of $\frac{1}{8}s$ for the *close-up*. We also capture the images using the smarphone flash lighting, with exposures of $\frac{1}{40}s$ and $\frac{1}{50}s$ for the *full size* and *close-up*, respectively.

5.A.5 Comparisons with Previous Work

We perform every comparison with previous work on an RTX NVIDIA 2080, using the default configuration for every method. However, for [Shi+20], we initialize the material graph with a fabric material (fabric suit vintage) provided in their repository, to better match our test data. For [Hen+21], we use the *fine-tuning* configuration.

Figure 5.11: A full diagram of our generator, including layer sizes and output dimensions for each layer. For Self-Attention, we leverage Linear Attention [Wan+20b], we use a MobileVIT transformer on the bottleneck [MR21], Group Normalization [WH18] and SiLU [EUD18] non-linearities, one decoder per output map and residual connections in every convolutional block. In red, we show the input/output dimensions (spatial, channels) of each layer; in orange, we show attention modules; in blue, convolutional blocks and layers; in green, upsampling and concatenating operations; in yellow, normalization layers; and in purple, regularizations, and non-linearities.

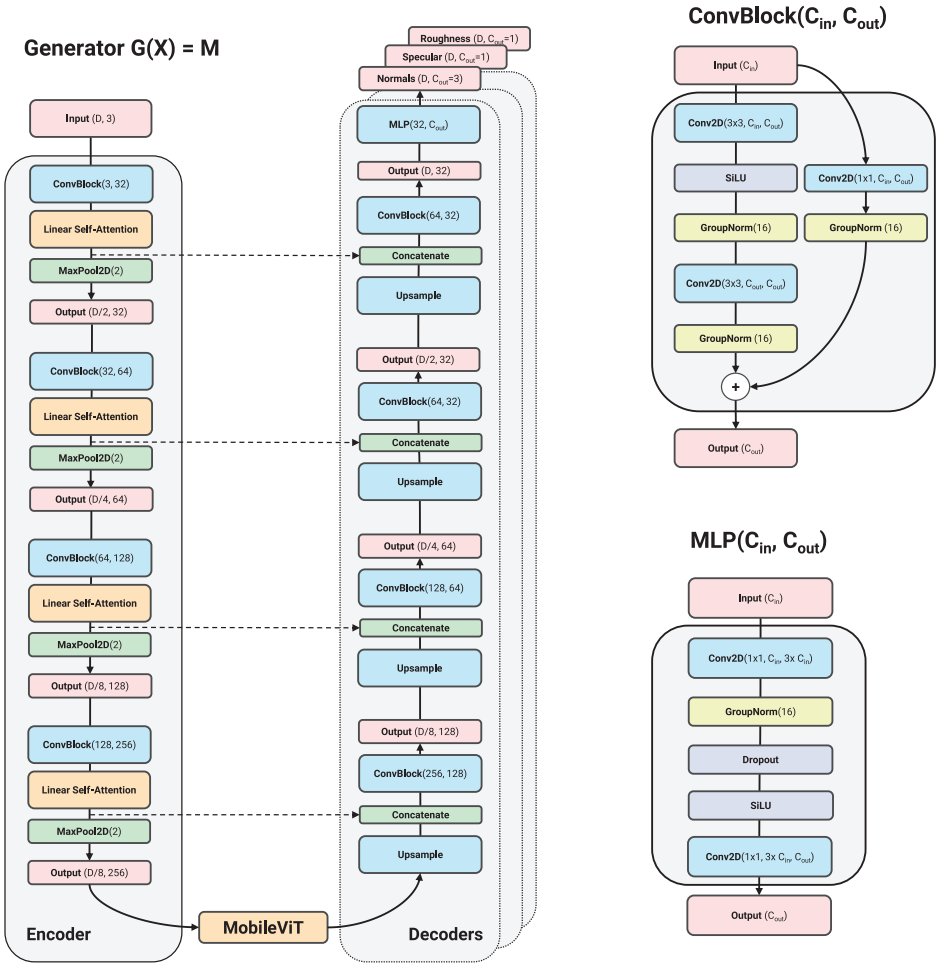
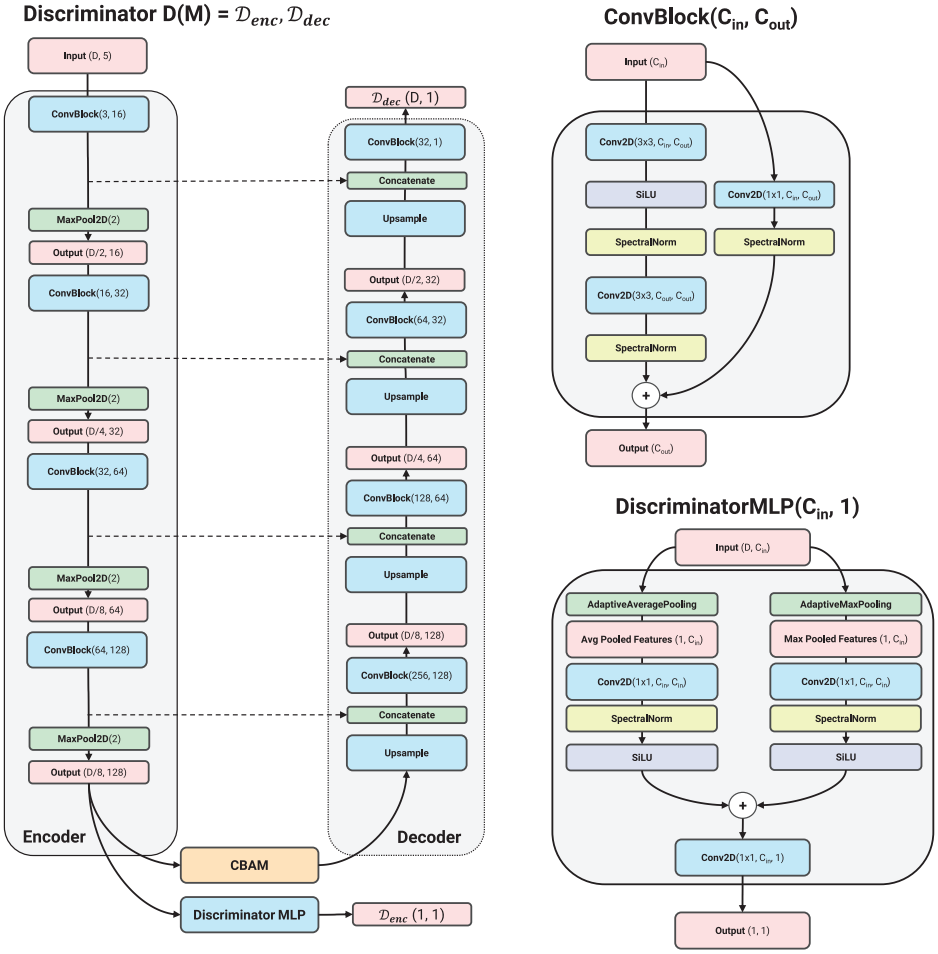


Figure 5.12: A full diagram of our U-Net residual discriminator, including layer sizes and output dimensions for each layer. We use a CBAM [Woo+18] module on the bottleneck and Spectral Normalization [Miy+18] throughout the network and residual connections in every convolutional block. In red, we show the input/output dimensions of each layer; in orange, we show attention modules; in blue, convolutional blocks and layers; in green, upsampling and concatenating operations; in yellow, normalization layers; and in purple, non-linearities.



5.B Dataset Analysis

Figure 5.13: Visualization of our dataset. We show the percentages of materials in our training dataset, including more detailed subcategories. On their right, we show the average specular and roughness for every category. As shown, there are some structures with distinct characteristics: Satins are highly specular due to the particularities of their yarns, and Piles (eg corduroy) or Plain Weave (eg linen fabrics) are much less glossy. We exploit this relationship between microgeometry and specularity for our estimations.

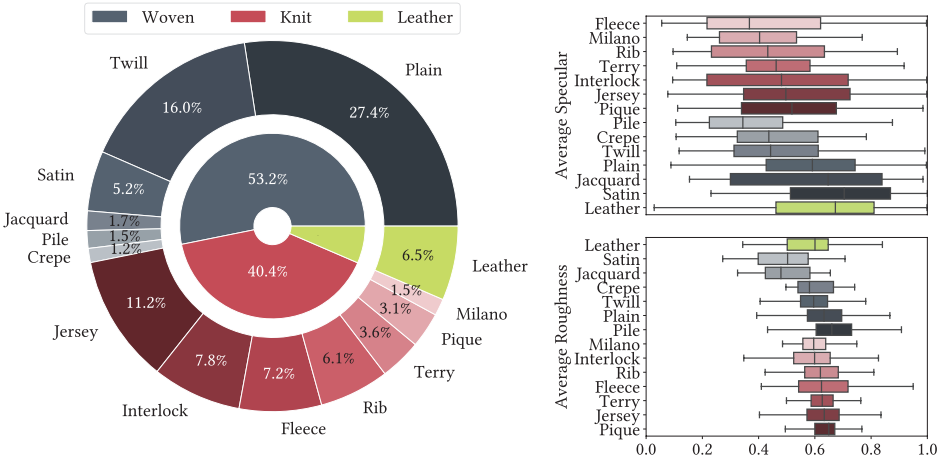
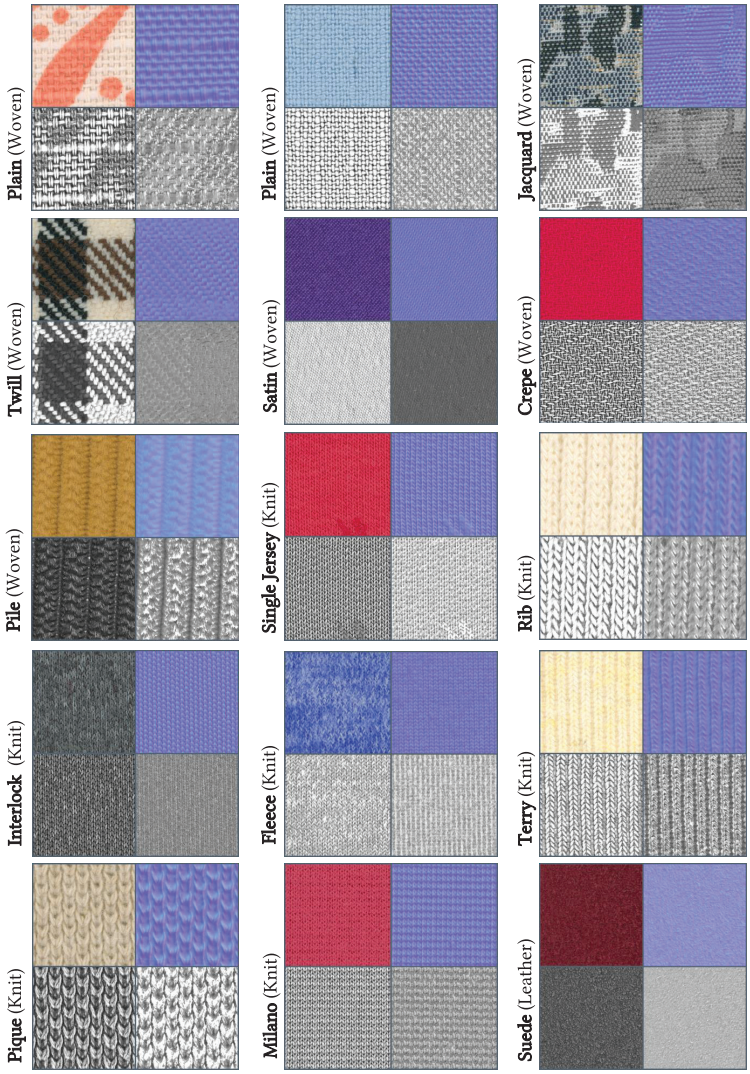


Figure 5.14: Visualization of some Ground Truth SVBRDF of the different families in our test set. Textiles have very complex and varied microstructures which play an important role on their appearance at different scales.



5.C Additional Details And Results

Figure 5.15: Additional results of our uncertainty quantification method. On the top, we show a beige *rib* fabric with gray metallic yarns. While there is a low uncertainty for the beige yarns, the metallic yarns are harder to digitize for our model (we do not support metalness in our material model) and it shows a higher uncertainty on those yarns. In the middle, we show a leather material with a very strong structural pattern. Our model shows very low confidence for this material. In the bottom, we show a tartan fabric. Interestingly, our model shows a higher uncertainty on the blue yarns, which are less common than the other yarns in the material.

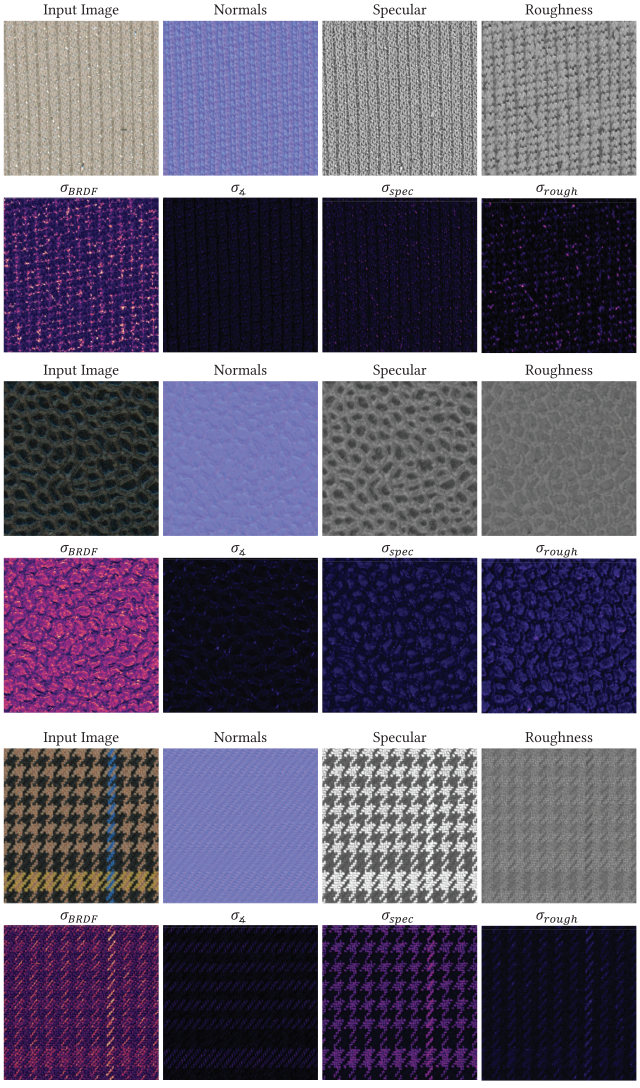


Figure 5.16: Additional results of our uncertainty metric. On the top row, we show a plot between our proposed render uncertainty σ_{BRDF} and the render error, which are highly correlated. We also show average error and uncertainty per family, as well as renders with the lowest and highest errors, compared to the ground truth. Below, we show the same data for normals, specular and roughness errors and uncertainty. There are no correlations between uncertainty and errors for these maps.

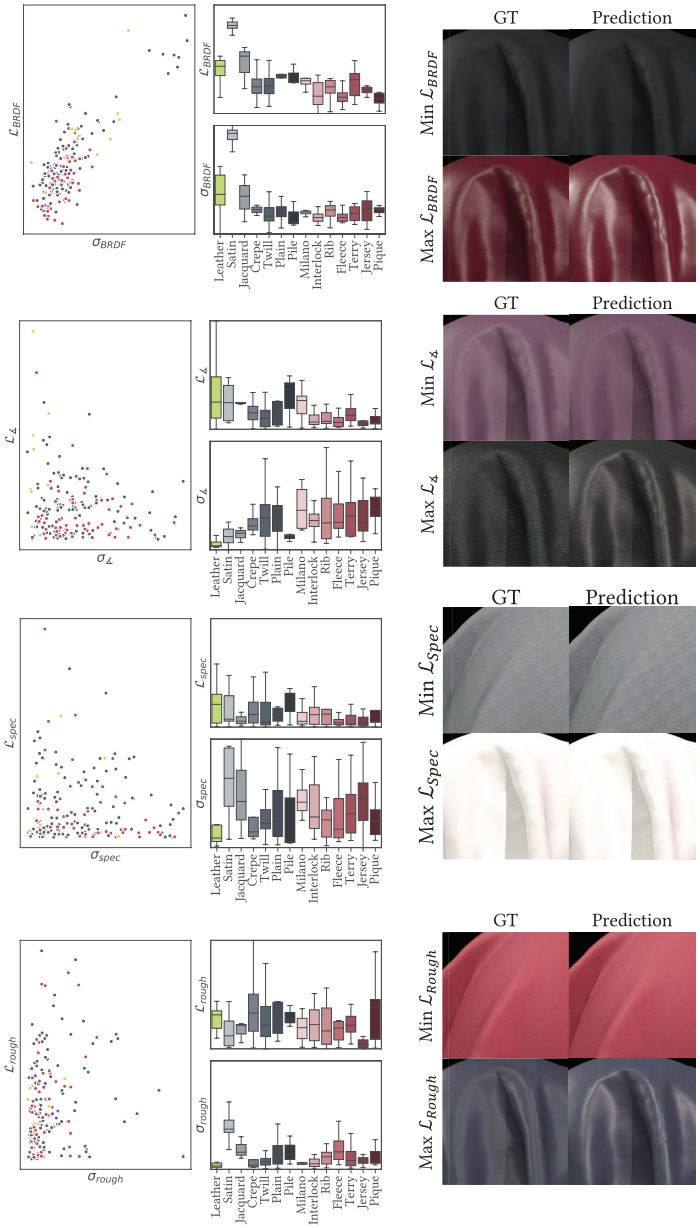


Figure 5.17: Additional results of our active learning experiment. From top to bottom, we show the renders of ground truth SVBRDFs, the model trained on the 100% of the training data, a model trained on 40% of the training data available, selected following an active learning approach using our uncertainty σ_{BRDF} as guidance, and a model trained on 40% data, selected randomly. From left to right, we show a very diffuse *Chiffon* fabric, a highly specular *Shantung* fabric and a *Goat Leather* material with very varied microgeometry. In every case, our model trained on 40% of the data following an active learning approach obtains results which are very similar to a model trained on 100% of the data. The model trained on randomly selected 40% data produces highly inaccurate specularity and microgeometry estimations.

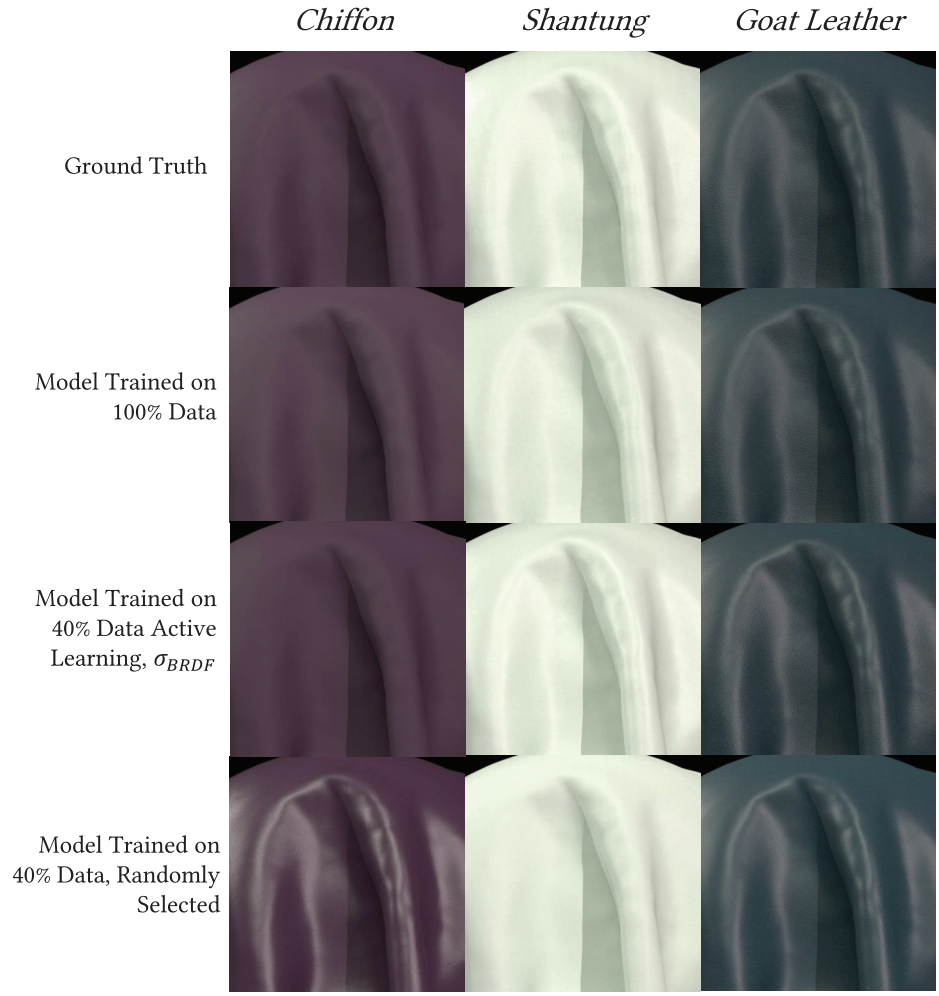


Figure 5.18: Results of our method on datasets of previous work. On the top, we show the results of [Hen+21] on their own test set. Our method provides sharper normals which better preserve the structure of the input images. In the middle, we show the results of our method on synthetic rendered data from a material graph from [Shi+20]. Our method provides sharp normals for this synthetic image, which lies outside the distribution of our dataset, composed exclusively of real images. On the bottom, we show an albedo and normals computed using Photometric Stereo [Ike81] for a captured BTF [WGK14], for which our model also provides highly detailed results.

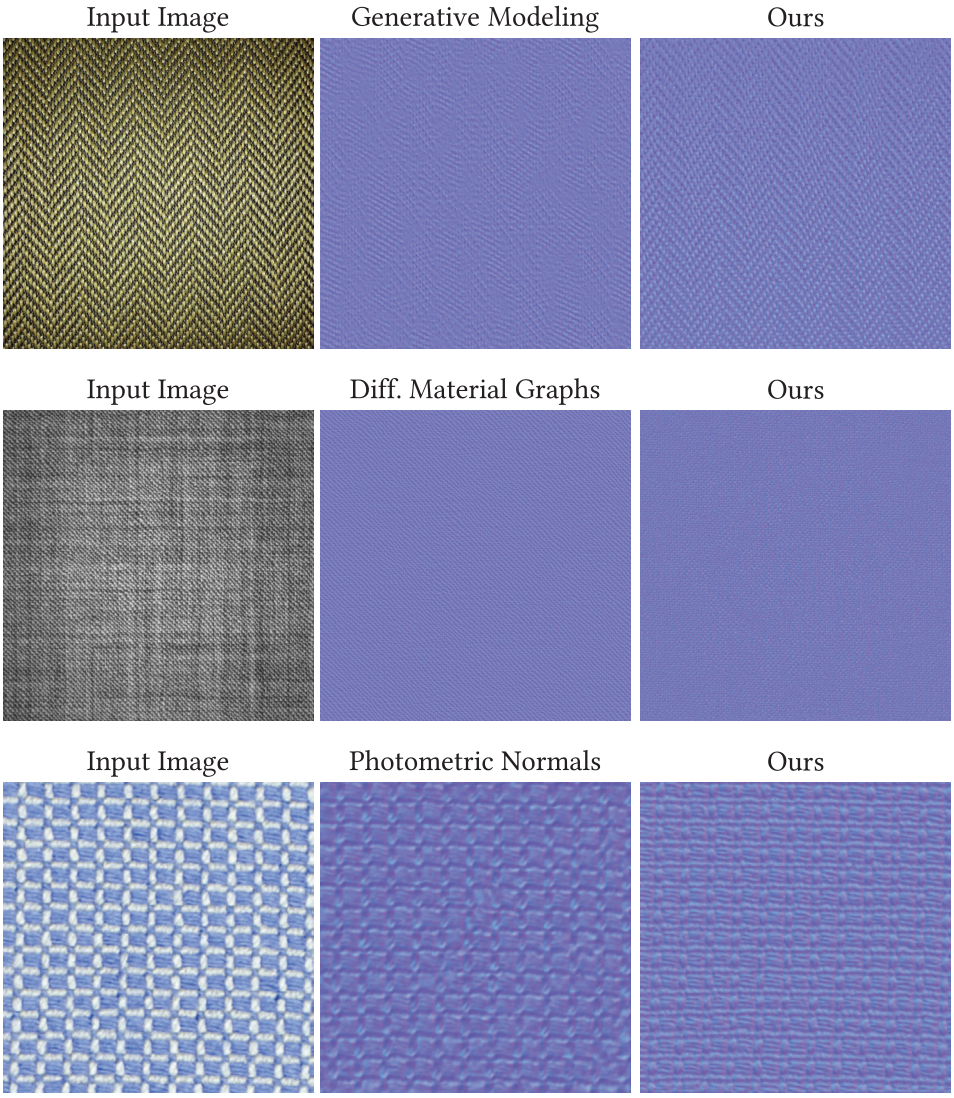


Table 5.4: Comparisons of our results with previous work on images captured under different conditions for a *curdoroy fabric*: On the first rows, images were captured with a smartphone using the flash image. On the middle rows, using the same smartphone with ambient lighting on different scales. On the final row, a scanner image. Note that for [Shi+20] we use a fabric material for initialization and use their metallic map instead of specular, that we do not estimate albedos and that the material models are not necessarily comparable.

Input	Deep Inverse R. [Gao+19]	Generative Model [Hen+21]	Diff. Material Graphs [Shi+20]	Adversarial Est. [ZK21]	**Our Method**

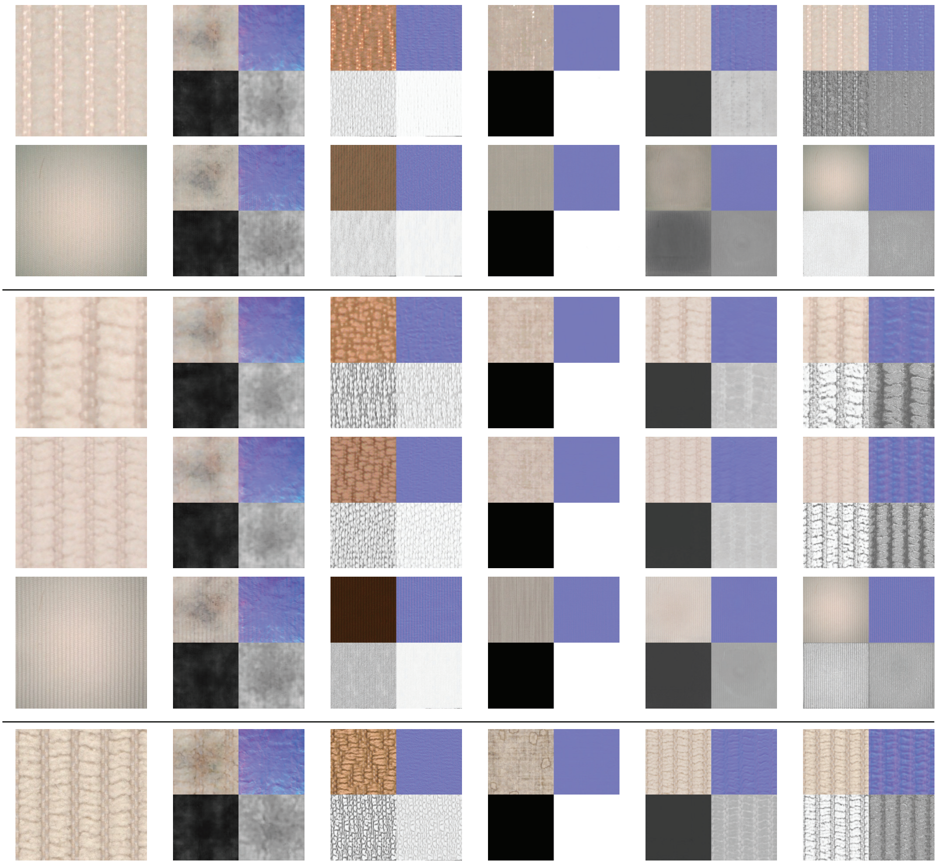


Table 5.5: Comparisons of our results with previous work on images captured under different conditions for a *suede leather*: On the first rows, images were captured with a smartphone using the flash image. On the middle rows, using the same smartphone with ambient lighting on different scales. On the final row, a scanner image. Note that for [Shi+20] we use a fabric material for initialization and use their metallic map instead of specular, that we do not estimate albedos and that the material models are not necessarily comparable.

Input	Deep Inverse R. [Gao+19]	Generative Model [Hen+21]	Diff. Material Graphs [Shi+20]	Adversarial Est. [ZK21]	**Our Method**