

CHAPTER 2. VISUAL ATTRIBUTE TRANSFER WITH PHOTOMETRY-GUIDED NEURAL NETWORKS

Figure 2.1: Illustration of the capabilities of our attribute transfer framework. On the left, we show a subset of our *photometric dataset*, comprised of several images of a material captured with different illumination conditions. On their right, two visual attributes: surface normals and segmentation maps. Using a larger image of the same or similar materials as *guidance*, our method can propagate the visual attributes, achieving high resolution, robust and controllable transfers at small computational cost.



This chapter presents a deep learning-based method capable of propagating spatially-varying visual attributes of a material (e.g. texture maps or stylizations) to larger samples of the same or similar materials. By leveraging the material's photometric response and a comprehensive data augmentation policy, we make the transfer robust to novel illumination conditions and affine deformations of the underlying surface. Our model relies on a supervised image-to-image translation framework and is agnostic to the transferred visual domain; we showcase a semantic segmentation, a normal map, and a stylization. Following an image analogies approach, the method only requires the training data to contain the same visual structures as the input guidance. Our approach works at interactive rates, making it suitable for material edit applications. We thoroughly evaluate our learning methodology in a controlled setup providing quantitative measures of performance. Last, we demonstrate that training the model on a single material is enough to generalize to materials of the same type without the need for massive datasets. The contributions presented in this chapter have led to the following publication¹, which was also presented at the 2022 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D 2022) as an invited talk:

¹Additional publication details and results are included on [the project website](#).



"Neural Photometry-guided Visual Attribute Transfer"

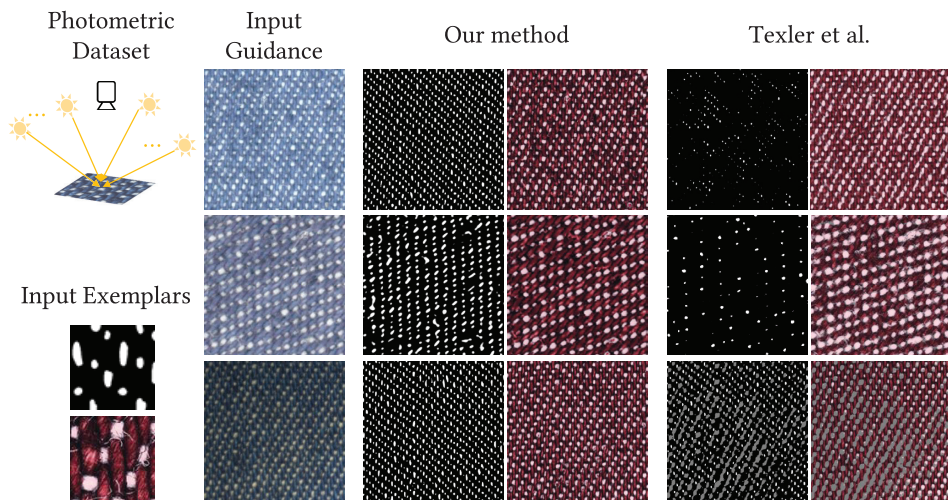
Carlos Rodriguez-Pardo, Elena Garces

IEEE Transactions on Visualization and Computer Graphics (TVCG)

(2021)

2.1 Introduction

Figure 2.2: The input exemplars on the left-bottom are transferred to the input guidances (second column) using images of the material taken under multiple illuminations (photometric input) as training data. Using this data for training, Texler *et al.* [Tex+20b] fail to generalize to geometric distortions and illumination conditions not present in the training set.



The development of effective and editable material models is becoming increasingly important so that users of virtual prototyping, video-games or AR/VR applications can have compelling and realistic experiences. For instance, allowing for the creation of virtual environments that surpass the uncanny valley, or empowering artists to create breathtaking visual settings. Effective material representations require understanding the properties that uniquely define them, which we refer to as *visual material attributes*. These attributes are spatially-varying parameters that maintain spatial coherency with respect to the material structure while remaining invariant to changes in the scene illumination or the geometry of the

underlying object. For example, they may represent optical properties of a microfacet spatially-varying BRDF (albedo, normals, roughness, anisotropy, etc.), but also artistic stylizations or higher-level properties, as in semantic segmentation masks. The former can be obtained with specific optical capture devices and estimation methods, either using machine learning [Mer+19] or classic optimization methods [Ste+14]. The latter may require time-consuming human labor.

Obtaining these attributes for large material samples can be problematic in certain situations. For instance, measuring high-resolution surface normals of large material samples or generating densely annotated segmentation masks, require either unfeasibly large capturing setups or expensive manual input. A common strategy to handle these cases consists of propagating these attributes using a larger image of the same material used as guidance image. This problem has been formulated within the context of image analogies [Her+01]. Propagating, or transferring, these properties requires preserving the local spatial regularities of the material, as well as adapting to the global variations of the guidance image, which is a highly challenging problem.

Previous work has used PatchMatch-based synthesis [Mel+12], look-up-tables [RPG16], or neural networks [Maz+19] by looking for repetitive patterns, match color statistics, or image gradients. These methods are however prohibitive due to their runtime performance, taking hours. Image analogies methods have been extended so as to leverage the power of convolutional neural networks as image descriptors [Lia+17; Ben+21], but those methods do not yield predictable outputs, and their computational complexity yields them impractical in interactive applications. Recently, neural networks have proven successful for the task of video stylization [Tex+20b], where the user needs to provide a few input editing exemplars which are propagated at interactive rates to the rest of the video. We show that this approach lacks the generalization power to work with images taken under unseen resolutions and geometric or illumination conditions.

In this chapter, we propose a novel learning-based method to propagate any visual material attribute *estimated locally* for a material's to larger samples of it. We train a neural network per material using image-to-image translation methods, making use of a policy of data augmentation that makes the transfer invariant to affine transformations (scale, rotations and shears). As opposed to other methods which blindly apply data augmentation strategies, our method is robustly evaluated using a real dataset to guarantee predictability in the estimations. Further, illumination invariance is obtained by feeding the network with multiple images of the material taken under a diverse set of illuminations. Our models can be trained in less than a minute and generalize to materials with the same microstructure.

In summary, we present the following contributions:

- The first method to use photometric data to train an image-to-image translation model capable of propagating any kind of visual material attribute to larger samples of the material, regardless of the illumination conditions of the input image.
- A data augmentation policy, thoroughly evaluated with a real dataset, designed to make the transfer invariant to affine deformations.
- Exhaustive comparisons with related work, demonstrating that we can achieve more predictable and higher-quality mappings with a fraction of the computational cost.
- We show that our trained models generalize to materials with similar microstructure as the ones used for training.
- On Section 2.C, we show an extension on this method for High Resolution SVBRDF capture, which is part of a publication accepted to *SIGGRAPH 2023*.

2.2 Related Work

Several computer graphics and vision problems are closely related to our method. The most similar ones are those related to any form of by-example *visual attribute transfer* (e.g. color, texture, style, or geometry). Besides, we also review material estimation and capture methods.

2.2.1 Visual Attribute Transfer

Visual attribute transfer refers to the problem of transferring some visual attributes (e.g. color, style, texture, or geometry) of one or many exemplars to another exemplar while preserving its *content*.

This problem can be formulated within the context of *Image Analogies* [Her+01], in which the goal is to stylize a target un-stylized image B , given a pair of images A (un-stylized) and A' (stylized). The most common approach to tackle this problem has been via patch-based texture synthesis [Bén+13; Bar+15; Jam+19]. More recent approaches have leveraged the capabilities of deep latent spaces within convolutional neural networks to disentangle style from content [Ree+15]. A seminal work by Gatys *et al.* [GEB15b] utilizes a VGG-19 convolutional neural network [SZ15] pre-trained on ImageNet [Den+09] as a feature descriptor for images, in which style and content are associated with different layers of the network, and transferred by gradient descent optimization. Their work on *style transfer* has been extended for single images [HB17; Li+17b; JAF16; Che+17b] and video [Che+17a], as well as for developing image-space distance metrics which resemble human perception [Zha+18]. A shared limitation of many of these

methods is their narrow capabilities to provide predictable edits, and considerable focus nowadays is put towards this end [Gat+17; Gu+18]. A comprehensive review on the topic of neural style transfer is provided by Jing et al. [Jin+19]. Our work differs from traditional style transfer approaches in the sense that we deal with a more constrained problem that requires predictable outcomes.

Exploiting the power of deep neural networks in the image analogies problem was tackled by Liao *et al.* [Lia+17] who, by assuming a semantic prior over an exemplar input image and a target one, propose a method capable of finding a bijective mapping between both inputs, enabling two-way stylizations. Single-image generative models [SDM19] were extended to the image analogies problem in [Ben+21], by using convolutional neural networks to generate a new image with the style of an input *style* and the *structure* of another image. These methods, however, rely heavily on content or semantic features, making them vulnerable to lighting or geometric differences between the input images; and are computationally expensive, rendering them impractical for interactive applications. Our method is robust to both geometric distortions and illumination variations, and works at interactive rates. Similar in spirit to our method, as it explicitly considers texture variations due to illumination, is the work of Fišer *et al.* [Fiš+16], which applies patch-match to provide illumination-dependent exemplar-based stylizations to cartoon pictures. In contrast, our work is meant to be illumination-invariant. Also concerned with stylization problems, Texler *et al.* [Tex+20b] present a method highly related to ours. They apply patch-based training of an encoder-decoder deep neural network using key frames stylized by a user. Resembling our approach, their algorithm also follows a few-shot learning strategy using as training data a few exemplar patches. However, as opposed to our method, they do not account for the variability of the appearance of materials under different lighting and viewpoint so, as we show in Section 2.7, their method does not generalize to unseen illuminations or geometric variations.

Image colorization is concerned with colorizing a gray-scale image given a few colorized exemplars. In this problem, it is critical to infer semantic relationships between the images so that the new scene is perceptually coherent and plausible [He+18; Zha+19a; He+19]. Similarly, *edit propagation* methods [AP08; End+16] work by propagating strokes provided by a user to the rest of the image, removing the need for a semantic understanding of the input image. Our work is related to the latter techniques, as we leverage the feature spaces of the CNNs and do not require a large labeled dataset to effectively solve our problem, and it can also be used to propagate segmentation masks [LAA08].

2.2.2 Textured Materials

Many real-world materials show spatial regularities, commonly referred to as *textures*. The patterns present in textures can be parameterized, which

allows for low-cost material capture or synthesis models. A way to model textured materials is through BTFs (Bidirectional Texture Functions) [Dan+99; LM01], a technique that uses multiple camera views and lighting angles to capture a dense sampling of the appearance of a material. Inspired by such methods, we leverage several images of the material under different illumination conditions, however, we require fewer data than typical BTFs capture setups [Rai+19; Rai+20]. Similarly, the problem of extrapolating BTFs captures to larger material samples was addressed by Steinhausen et al. [Ste+15a; Ste+15b], who propagate measured BTFs using texture synthesis. Our method is not meant to propagate full BTFs measurements but could potentially be applied to such datasets.

The goal of texture synthesis is to reconstruct a larger image given a small sample leveraging structural content. This is a long-standing problem in the computer graphics field and different strategies have been proposed, for instance, using PatchMatch [Dia+15], texture transport [AWL15], point processes [Gue+20; LH06], or neural networks [EM17; Zho+18; FAW19; Rod+19]. Also related to our work, Li *et al.* [LPG19] capture the appearance of materials by first estimating their BRDF and, then, synthesizing the high resolution microstructure from a dataset of measured SVBRDFs. Our problem is unlike texture synthesis, as we do not aim to create novel content but to predictably transfer visual material attributes.

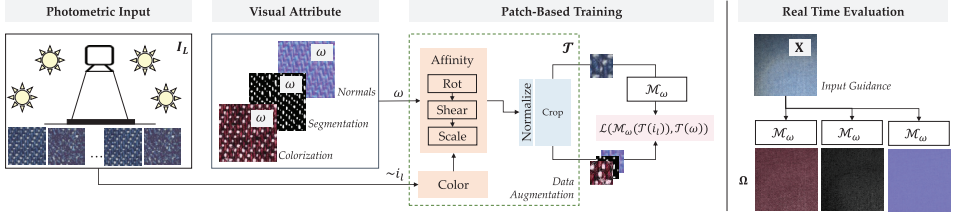
2.2.3 SVBRDF Estimation

The problem of estimating a SVBRDF model from one or several images using lightweight capture setups is becoming increasingly popular in the literature. Early work [HS05; HS03] leveraged Photometric Stereo [Ike81] and SVBRDF manifold bootstrapping [Don+10] for surface geometry reconstruction, while newer methods exploit the power of deep neural networks. Recent surveys by Guarnera [Gua+16] and Dong [Don19] contain relevant approaches. While our method is not meant to estimate the SVBRDF properties of a material, it can be used in combination with those techniques to create larger material assets.

There are a few methods that follow a similar paradigm to ours, transferring pre-estimated SVBRDF maps to a larger material sample. Using PatchMatch texture synthesis, Melendez *et al.* [Mel+12] transfer displacement and albedo maps from small samples of the materials. Their method is limited to daylight illumination and materials present in façades. By means of look-up-tables, and using surface normals and specular as guidance, Riviere *et al.* [RPG16] transfer surface reflectance captured with controlled LCD lighting to a material sample observed under natural lighting. Recently, Deschaintre *et al.* [Des+19] fine-tune a network trained to estimate SVBRDFs [Des+18], to work on larger material samples taking a guidance image as input. This approach is limited

to transfer a pre-defined set of property maps while our method can transfer any kind. The strategy of using multiple images of the material under different illuminations as input data is not new. Li *et al.* [Li+17a] and Ye *et al.* [Ye+18] utilize a self-augmentation strategy to make the estimation of the SVBRDF more robust to unknown environment illumination. We are inspired by these approaches to increase robustness in the model predictions.

Figure 2.3: Overview of the method. We learn a mapping between the photometric response I_L of the material and a visual property map ω . We make this mapping robust to affine transformations by means of a particular policy of data augmentation used for training. We learn one model M per material and visual attribute, which allows us to robustly evaluate the performance of the method under several transformations of the guidance images X . At evaluation time, M can have any size. The training of M per visual attribute w takes less than a minute.



2.3 Problem Formulation

Our goal is to transfer a D -dimensional spatially-varying *visual attribute* w of a material (for example, estimated locally at high resolution) to a larger sample of it.

We formulate the problem with an image-to-image translation approach. For training, our method takes as input: a *photometric* dataset I_L , and a *visual attribute* ω . The *photometric* dataset consisting of a number of RGB planar images of a material, $I_L = \{I_l | I_l \in \mathbb{R}^{n \times m \times 3}, |I_L| \geq 1\}$, illuminated with different light sources $l \in L$, of $n \times m$ pixels. This kind of images can be either captured with specific devices [Nam+16; MWK17; Alc+19], or synthetically rendered given an inverse material estimation pipeline [Guo+20b; DDB20]. The visual attribute being a spatially-varying map $\omega \in \mathbb{R}^{n \times m \times D}$ of any kind, and dimensions D , that maintains pixel-wise correspondence with the photometric input images I_L . Figure 2.3 and Figure 2.5 show examples of these images for three different visual attributes: a stylization, a segmentation, and a normal map.

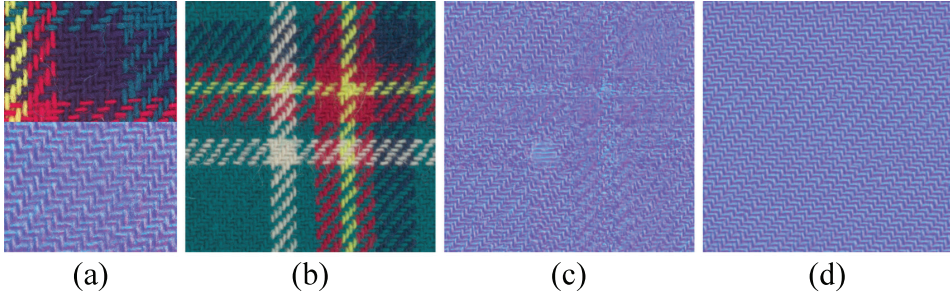
Given this data for a single material, and a strategy of patch-based training, we learn a function M that can be applied to a new *guidance* image of the input material (or a similar one) $X \in \mathbb{R}^{N \times M \times 3}$ of any size $N \times M$, to get its corresponding visual attribute $\Omega \in \mathbb{R}^{N \times M \times D}$.

$$\mathcal{M}: X \rightarrow \Omega \quad (2.1)$$

$$\text{s.t. } i_l \xrightarrow{\mathcal{M}} \omega, \forall i_l \in \mathcal{I}_L \quad (2.2)$$

At evaluation time, the guidance image X might contain different colors, scales, illumination, or affine distortions than the images used for training. Figure 2.3 shows an overview of the training and evaluation processes. In Section 2.6, we evaluate the conditions of the input guidance image upon which the method provides robust estimations.

Figure 2.4: The color augmentation policy takes advantage of the material structural regularities to make the transfer robust to different albedos. (a) A diffuse image of the portion of the material used for training, and its corresponding normal map. (b) Input guidance image. (c) Transferred normals without the color augmentation policy. (d) Transferred normals using color augmentation. Note that the training data does not include images containing the white yarn.



2.4 Learning Framework

In this section, we describe the patch-based and data augmentation strategies used for training, the neural network design and loss functions, and the implementation details.

2.4.1 Patch-based Training

Each training step takes as input a pair of corresponding image patches taken from the photometric input I_L , and the visual attribute ω . Using this data alone already provides a good starting point for generalizing to unseen illumination setups. However, it is not sufficient in scenarios in which the guidance image contains variations due to image noise, a different scale, or any other affine distortion. In order to make the transfer invariant to these transformations, the network needs to be trained with the appropriate data. Data augmentation strategies are essential for reducing the amount of necessary data for training

[SK19; Kar+20a], however, random strategies not taking into account the particular domain might degrade the quality of the prediction. We therefore follow a pre-defined data augmentation policy T (illustrated in Figure 2.3), where random operations are performed sequentially.

Color Augmentation Even if the microstructure of the material is homogeneous and can be measured using only a small patch of it, it may not be possible for the network to estimate its visual attributes for parts of the material which contain previously unseen colors. We correct this by randomly permuting the color channels of the photometric input (see Figure 2.4). As we show in Section 2.7, this data augmentation policy also helps make the model generalize the transfer to similar materials, by learning features that are more related to the structure of the material than to its color. Similar operations have been recently proposed for finding robust visual representations on self-supervised settings [Che+20b]. Only the photometric input is subject to this transformation.

Affine Transforms In order to allow for material editing applications in real images, the model should generalize to images taken under camera perspectives and geometry distortions different than those present in the fronto-planar images used for training. Many of such texture irregularities can be defined as an affine transform: translation, rotation, shear, or scaling. As CNNs are shift-invariant by design [Kau17], we propose to augment our datasets with random transformations for scalings, rotations, and shears. Those transforms can be efficiently performed on images through matrix multiplications. However, some visual property maps need to be treated specially, as the spatial transform might have a different behavior in 3D vector space, e.g. normals or tangents. In those maps, each pixel is a representation of a 3D vector. As such, we also perform the rotations and shear operations to these maps in 3D space, by multiplying each normal vector by the same affine transformation matrix applied to the 2D image. We perform the rotation around the Z axis, thus, assuming the camera sensor is parallel to the object plane.

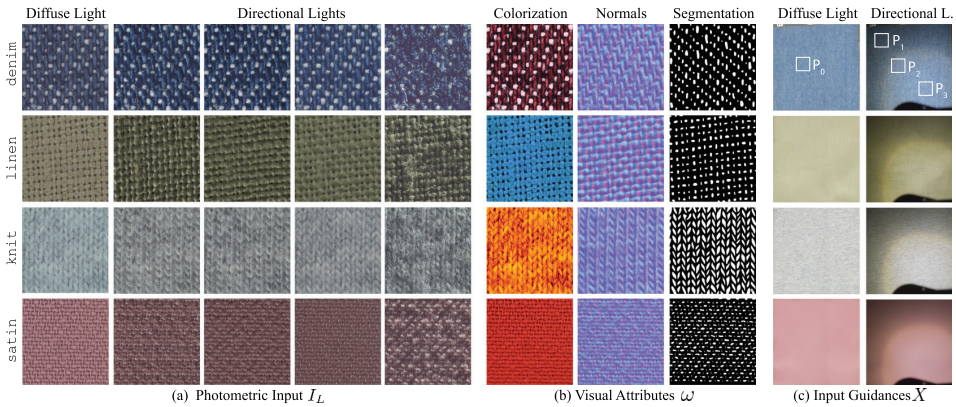
Cropping Inspired by recent work on patch-based learning [Tex+20b; Par+20], during training, the network receives small patches of each input pair. Those patches are randomly cropped from the randomly augmented images, so the network receives a considerable amount of variations of the same material, thus making generalization possible.

2.4.2 Network Design

We follow a uni-modal image-to-image translation learning strategy, assuming there is only one correct mapping from input to output image. This approach is reasonable for the kind of transfers we test in this work. However, it might

fail for ambiguous cases where there are multiple suitable outputs for the same input, for which multi-modal approaches [Zhu+17b] or GANs are more advisable, albeit harder to train. Specifically, our model M is a shallow U-Net network [RFB15] with 4 blocks of layers, containing a small number of trainable parameters, inspired on few-shot learning strategies [Wan+20d; Wan+19; Liu+19b]. This type of fully convolutional architecture is of common usage in image-space regression problems, due to its capability of efficiently learning patterns at different levels of abstraction thanks to its multi-scale design. Skip connections are added to enhance local details [RFB15; Iso+17]. The small number of parameters allows for faster training and inference, as well as reduced memory usage.

Figure 2.5: An overview of our evaluation dataset. (a) Five example images of our high resolution captured data illuminated with diffuse light and four directional light sources. (b) Examples of some of our *visual property maps*: colorization, normals and yarn segmentation. (c) Evaluation images taken under diffuse and directional illumination sources. *Denim*, (c) contains labels for several patches used for the evaluation in Section 2.6. Those materials show different properties that may prove challenging for our visual attribute transfer task. Our *denim* and *knit* fabrics show diverse color variations (different colored dyed yarns and stochastic albedo, respectively), *linen* shows strong geometric variations and *satin* shows anisotropic optical behavior.



Note that, as opposed to previous work [DDB20] on material transfer, which is initialized from a pre-trained network, but inspired by single-sample image synthesis methods [Zho+18; SDM19], we train one network per material and visual property. This strategy, although increasing training time, is key for the following reasons: First, it allows us to better understand the generalization capabilities obtained through our data augmentation policies. Second, it guarantees predictability of the trained model as every feature learned by the network is specific for each material and visual property pairs. Finally, it removes potential problems of a biased dataset as there is no cross-material

or cross-domain learning. If the input dataset does not contain enough variations of the material to represent the whole material, the model will fail in areas with unseen patterns. In those cases, having a pre-trained network may help as a material prior, as in [DDB20]. However, our method obtains comparable results to pre-trained methods, with a smaller computational footprint and with the additional flexibility of not needing an expensive dataset and large models. In practice, training a single network per material and attribute is not problematic as this process takes less than a minute.

Loss Function Choosing the appropriate loss function for a learning framework highly depends on the problem. For example, some methods [Iso+17] combine a per-pixel L_1 metric with adversarial [Goo+14a] losses, as the latter allows for better semantic mappings in multi-modal learning scenarios, whilst the former allows for improved predictability. Texler *et al.* [Tex+20b] further includes a perceptual loss [Tex+20a], while using a render loss is common in methods that estimate material parameters from photos [Des+18; DDB20]. In our method, we show that using a L_1 loss for training is enough for learning accurate and predictable mappings in our regression task, and binary cross-entropy loss for semantic segmentation, following standard practice in image segmentation [RFB15]. We found that the L_2 loss function yields overly-smooth outputs, and perceptual loss functions like $LPIPS$ [Zha+18] are more prone to artifacts than pixel-wise ℓ_{norm} metrics. We include an ablation study of the impact of the loss function in the supplementary material of this chapter.

2.4.3 Implementation details

We use PyTorch [Pas+19] as the learning framework, Adam [KB15] for optimization, a learning rate of 0.002, and a batch size of 16. The training images are randomly augmented using uniform distributions by the following operations, in order: First, the photometric input is subject to the color augmentation policy. Then, both inputs and targets are randomly rotated by an angle in the $[-90, 90]$ range, randomly sheared by an angle in the $[-45, 45]$ range, and randomly rescaled in the $[0.5, 2]$ range of scale factors. Then, patches of 128×128 pixels are randomly cropped during training to generate a large dataset of images.

All inputs are always standardized using their own mean and variance. Each M is trained for 1000 iterations, which takes around 1 minute on a single Nvidia 1080Ti GPU. Due to the fully convolutional nature of our models and their reduced number of trainable parameters, the guidance images X used for evaluation can be of arbitrary dimensions. We used images of up to 5000×5000 pixels for which evaluation takes around 150ms. We refer the reader to

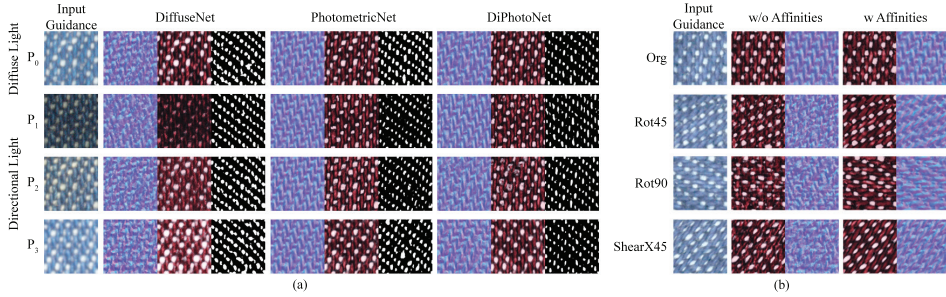
the supplementary material in this chapter for a comprehensive description and a diagram of this model, as well as further implementation details.

2.5 Dataset and Metrics

2.5.1 Dual-Resolution Captured Data

Our method is agnostic to the capture setup [Nam+16; MWK17; Alc+19], and it may work for any kind of input data (e.g. BTFs [LM01; Rai+19]) as long as the photometric images are pixel-wise aligned with each other and to the visual attribute. This dataset could also be created synthetically by rendering the outputs of any SVBRDF estimation method [DDB20]. In Section 2.7, we show results of our method using these acquisition pipelines.

Figure 2.6: Qualitative results of our models under different datasets and data augmentation configurations, for different inputs of the *denim* material. On the left (a), we show the results of our networks trained using different dataset configurations, under a guidance image taken with diffuse lighting (p_0), and three crops of a guidance image illuminated with a directional light (p_1, p_2, p_3) not present in the training set. Please refer to Figure 2.5 for the position of these crops on the larger guidance images. On the right (b), we show the results of our *photometricNet*, under different geometric distortions (rotations and shears) performed to its guidances images, taken under diffuse illumination.



However, for the purpose of this evaluation, we choose to work with real data. The main reason is that the data obtained with material capture devices poses extra challenges that are difficult to reproduce with render engines. For example, material irregularities, complex optical behavior, or distortions and color shifts introduced by the optical system that might cause the models to produce inaccurate estimations.

We create a dataset containing images of the same material taken with two imaging camera systems: a high-resolution camera that allows us to take pictures of 0.7×0.9 cm, with a resolution of 367×490 pixels and a macroscopic camera which provides images of 11×11 cm, with a resolution of 4800×4800 pixels. In

terms of illumination, our setup has 27 different collimated light sources uniformly distributed across the hemisphere as well as diffuse illumination. We build a dataset of four different textile materials, whose complex optical behavior due to anisotropy, transmittance, directionality and microstructure [CLA19] turns them particularly challenging for synthesis and editing operations [Kam+16; Rai+19]. For training, we capture one image for each light source, making a total of 28 different images (Figure 2.5 (a)). For the evaluation set, we take one *guidance* image with diffuse illumination and another *guidance* image with a directional light source (Figure 2.5 (c)). The visual attributes (Figure 2.5 (b)) are generated automatically using photometric stereo [Ike81] in the case of normal maps and manually by artists in the cases of colorizations and segmentation masks.

2.5.2 Attribute-specific Metrics for Evaluation

For evaluating our models, we choose domain-specific distance metrics different from those they were trained with to better understand their generalization capabilities [TOB15]. For normal maps, we compute the cosine distance between ground truth and estimated maps, as it accounts for the geometric space in which normals lie. In the case of image segmentation, we evaluate the results using the *Jaccard similarity coefficient* (IoU) [Zho+19], which is well-suited for sparse segmentation tasks. Finally, we use the state-of-the-art metric *Learned Perceptual Image Patch Similarity* (LPIPS) to evaluate the quality of the colorizations, as it has been shown to outperform L_2 norm for visual perception tasks [Zha+18].

2.6 Evaluation

We evaluate our method in different settings. First, we assess the type and amount of photometric input data necessary for the model to generalize to different illumination conditions and image distortions. Then, we test our data augmentation strategy for affine transformations.

2.6.1 Invariance to Input Illuminations and Distortions

In this set of experiments, we aim to evaluate if the model produces the same output after changing the training data and input guidance images. To this end, we measure both the impact of different illuminations and sizes of the photometric dataset, as well as variations of the input guidance images.

Photometric Input

In this experiment, we study the type of photometric input that makes the method invariant to different illumination conditions of the guidance image. We

compare three models trained with different datasets: *diffuseNet*, which uses a single image illuminated with diffuse lighting; *photometricNet*, which takes as input 27 directional lights; and *diphotoNet*, which uses all the 28 sources. For evaluation, we use the macroscopic camera which captures a larger sample of the material at a lower resolution. We take two test guidance images with diffuse and directional lighting (Figure 2.5 (c)). In this experiment, all images are aligned, therefore, we follow a limited policy of data augmentation from Section 2.4.1, applying only color augmentation, rescaling, and crops, and leaving out rotations and shears for in-the-wild scenarios.

Figure 2.6 shows qualitative results of our method for a selection of patches for the *denim* material. Figure 2.6 (a) shows that the best accuracy is in general obtained with *diphotoNet*, i.e. training with both directional and diffuse illuminations. This is reasonable, as the model is trained using the same illuminations used during test time. Conversely, *diffuseNet*, a network only trained using diffuse lighting is less capable of generalizing under any kind of illumination source. This suggests that following a photometric approach for training material synthesis models allows for better generalization capabilities. Finally, the results for *photometricNet*, which is trained only with directional lights shows similar accuracy as *diphotoNet* while proving generalization to unseen illuminations. For all the results shown in the chapter, we have chosen *photometricNet* as our model. This has an additional advantage from the usability perspective: it is relatively easy and cost-effective to build a capture setup (such as the flash light from a smartphone) or generate renders with directional lights, while it is considerably harder to recreate both types of illumination consistently.

Dataset Size Influence

Our goal in this experiment is to understand how many images [NJR15] taken under directional lights are needed in order to obtain the desired invariance to illumination.

We train different models using reduced versions of our full 27-image dataset, and compare their results to those of the *photometricNet* trained on the full directional dataset. More precisely, we train networks using 1, 3, 9 and 18 directional lights for each material and application in our dataset following the same reduced data augmentation policy described in the previous experiment. Instead of randomly selecting light sources around the hemisphere, we perform a more sensible light source sampling. Specifically, for each reduced dataset, there is at least one light that is as close as possible to the normal of the surface on which the material lies, thus giving more importance to frontal angles. We extend these selected lights to 3 for the reduced datasets with more than three lights. The rest of lighting sources are uniformly sampled around the hemisphere, up to a zenith angle of 70 degrees. The supplementary material contains a diagram of the distribution of lights in the hemisphere. Results are shown in Figure 2.7, where we see

that adding more lights to the dataset monotonically increases the generalization of every model. However, adding lights consistently shows diminishing returns, which suggests that a capture setup with around nine lights might be enough for relatively accurate estimations. Similar findings are reported in multi-image SVBRDF estimation methods [Guo+20b; Des+19].

Figure 2.7: Error of the reduced *photometricNets* on our ground truth guidance image (taken with diffuse illumination, not present in the training dataset) for each material in the dataset and the three visual attributes and corresponding error metrics, which are close to the normal of the surface in which the material lies. The width of the lines indicates the standard variation across 5 repeated experiments, where different light directions were randomly chosen to form the training datasets. Please refer to the supplementary material for the position of each light source.

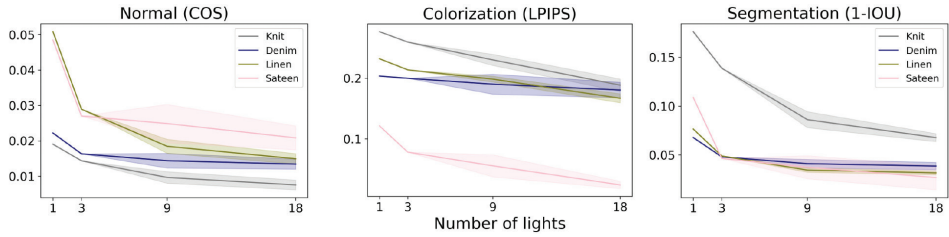


Image Degradations

As discussed in Section 2.5.1, real captured images may be subject to distortions, shifts and noise introduced by the optical capture system. To fully understand the robustness of our models with respect to these types of imperfections, we synthetically modify the saturation, contrast and noise present in the X guidance images. As shown in Figure 2.8, M is robust to these types of degradations, even when a fair amount of details are lost in X . These images were not part of the training dataset.

2.6.2 Equivariance to Affine Transforms

Our goal in this experiment is to analyze the equivariance of our model to affine distortions of the guidance images. A function f is said to be *equivariant* with respect to a transformation T if $f(T(x)) = T(f(x))$. Inspired by [Ben+20], we measure the equivariance of our model M with respect to different affine transforms T performed to an input guidance image X , by computing their difference using the corresponding metric defined in Section 2.5.2 $d(M(T(X)), T(M(X)))$. To achieve this, we extend the augmentation policy in the previous experiments by adding random shears and rotations to the training process. We train the *photometricNet* with and without random shears and rotations for every visual attribute in our dataset and on the *denim* material. In the case of the normals, as described in

Section 2.4.1, we also perform the shears and rotations in the geometric space in which normals lie.

We measure their robustness with respect to three different transformations T : rescalings, rotations and shears. As before, we use a guidance image taken under diffuse lighting as input for these experiments and attribute-specific distance metrics. Figure 2.6 (b) shows that not adding those transforms generates visual artifacts for rotations and shears. Interestingly, without affine augmentations, the models hallucinates vertical yarns, as it is the only type of data that it has seen as input. Figure 2.9 shows quantitative metrics for the range of transformations T in which we evaluated the models. Augmenting the training dataset with both shears and rotations generally provides the best results. Furthermore, this enhanced data augmentation policy improves the robustness of the models with respect to the scale of their inputs. Notably, applying random rotations during training appears to have a larger impact than random shears on the robustness of the models. This finding suggests that applying blind policies of data augmentation [ASE17; SK19; San+19; Che+20b] may not be an optimal strategy for some applications like ours, as the networks may not learn relevant features, or overfit to noise present in the training dataset. Finally, it is worth noting that every network has the same number of parameters, and is trained for the same number of iterations, which shows that the generalization capabilities of the models can be increased at no extra parameter cost.

Figure 2.8: Output of our model under different distorted inputs: a change of saturation, contrast, and Gaussian noise ($\sigma^2 = 255$). The number at the bottom is the cosine distance with respect to the original estimation. As we can see, the output is consistent with a very small error in all cases.

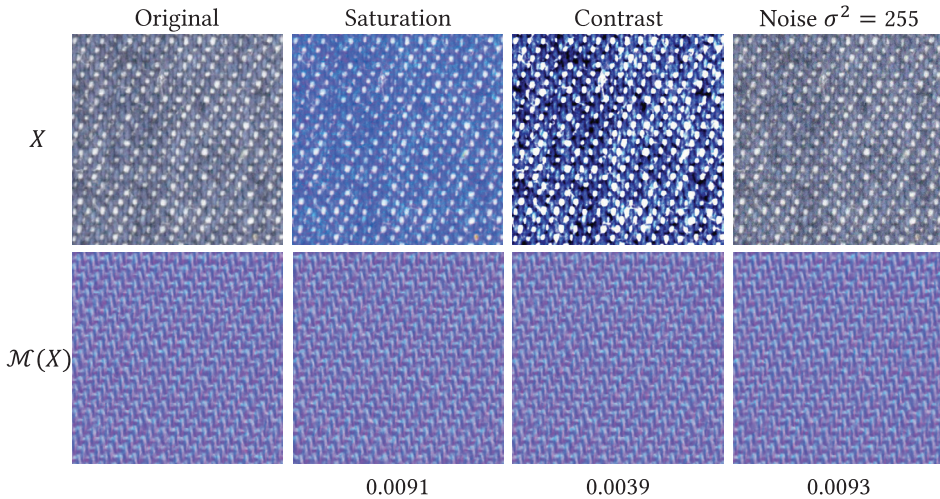
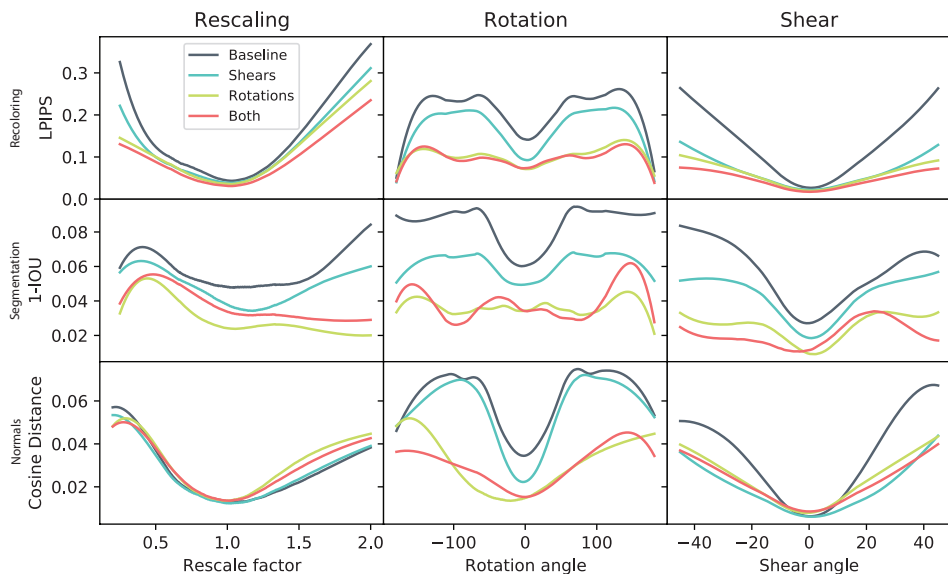


Figure 2.9: Quantitative evaluation of the robustness of our networks with respect to different transformations T (rescaling, rotation and shearing) of networks trained under different data augmentation policies for the *denim* material. From top to bottom, we show results on: recoloring, segmentation, and normals estimation. Lower is better for each metric



2.7 Results and Comparisons

In this section, we first compare our method with related approaches on image stylization and large scale SVBRDF material transfer. Then, we show the capabilities of our method to generalize to similar materials to those in their training set and present its limitations.

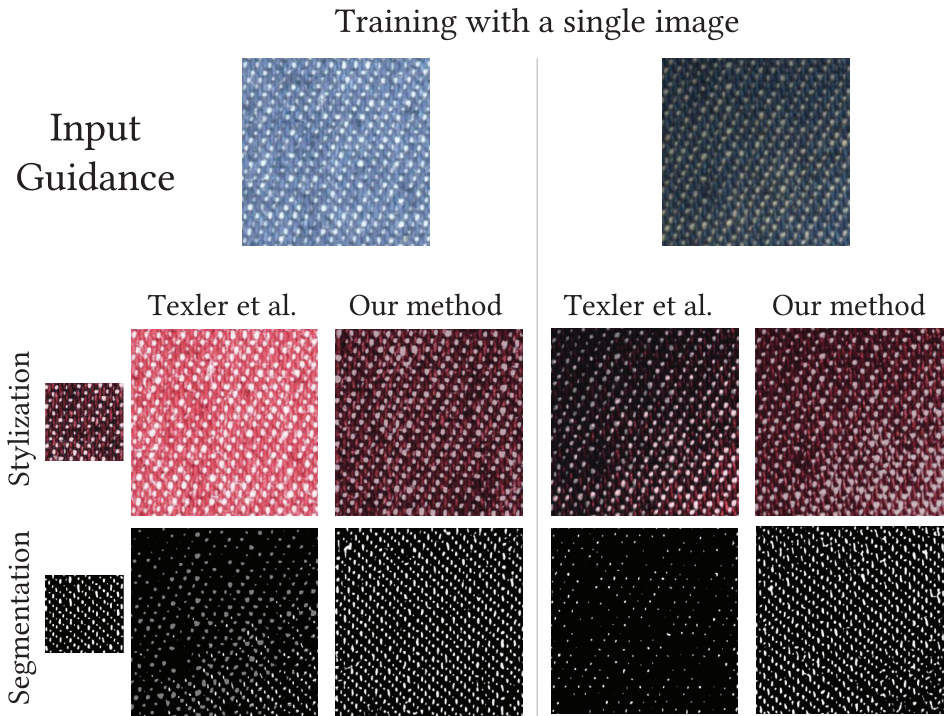
2.7.1 Interactive Stylizations

In the first category, the method of Texler *et al.* [Tex+20a] allows artists to interactively edit a few keyframes of a video and propagate that edition to the rest of the video. As ours, they formulate this transfer problem by training an encoder-decoder network using patch-based learning. But, in contrast to our work, they do not perform any data augmentation policy outside of random cropping. For a fair comparison with such method, we compare two setups: 1) using a single image as training input, and 2) using the photometric dataset. Results are shown in Figures 2.2 and 2.10. As Texler’s method is not scale invariant, in both cases the training data provided for their model has the same scale as the images used for testing.

Our models have been trained with the full policy of data augmentation. In the first setup (Figure 2.10), we use the diffuse illumination and hence compare their output with our *diffuseNet* output. As shown, none of the methods provide high quality results but our model manages to provide closer estimations. In the second setup (Figure 2.2), we train their model with the photometric input. The best results are obtained with our method. Even though extending the input data using photometric cues has an impact on the quality of Texler’s results, the lack of a data augmentation policy makes the transfer fuzzier and noisier. Further, their combination of style, adversarial and pixel-wise losses fails to yield predictable mappings. These results confirm the importance of a comprehensive data-augmentation policy, such as the one we propose, when using neural networks for image processing tasks of this kind. In addition, our model is trained in less time with a smaller computational footprint (1 minute vs 5 minutes).

Figure 2.10: Comparison of our method with Texler *et al.* [Tex+20b] using a single diffuse image of the *denim* material as training data. The task is to transfer the two attributes shown (*Stylization* and *Segmentation*) to two different guidance images.

Even using a single image instead of a photometric dataset, we achieve higher quality mappings at a lower cost.



Another way of formulating this visual attribute transfer problem is through *image analogies*. Using our single diffuse image for input, we compare our approach with two methods as shown in Figure 2.11. First, the work of Liao *et al.* [Lia+17], which uses deep latent spaces as image descriptors; and the method of Benaim *et al.* [Ben+21] that trains single-image generative models to find bijective mappings between the *structure* of one image and the *style* of another. Our method qualitatively outperforms these methods with a fraction of the computational cost: 1 minute in our case, 10 minutes in [Her+01], 40 minutes in [Lia+17] and 10 hours in [Ben+21]. Once trained, our models can be used to evaluate any guidance image in real time for materials with similar microstructure. In contrast, image analogies methods require expensive optimizations for each guidance image. We refer the reader to the supplementary material for more comparisons with these methods.

Figure 2.12 shows additional results of material stylizations. In these examples, we used the method of Gatys *et al.* [GEB15b] to stylize a small patch of the material. Then, we trained a model using photometricNet and diffuseNet. As the guidance image we used a bigger image with diffuse illumination. Compared with naïve style transfer applied to the whole image, our approach provides detailed stylizations where the microstructure of the material is preserved. We further see that diffuseNet provides noisier results than photometricNet, probably due to the fact that the photometric cues help to preserve the local shading variations.

2.7.2 Creation of Large Scale Digital Material Assets

Our method can be used to propagate SVBRDFs estimated locally in a small area of the material to larger samples. Figure 2.13 illustrates for a diverse set of materials that we can propagate albedo and normals estimated at high resolution in a small area of 0.7×0.9 cm to guidance images of 13×13 cm taken with a smartphone. Even though we train the albedo and normals models separately, which does not guarantee pixel-wise coherence between the estimated maps, the rendered images show realistic-looking materials, even with a diffuse material model. As opposed to the method of Deschaintre *et al.* [DDB20], which is trained to output directly Cook-Torrance [CT82] material layers, our method is agnostic to the parameters of the SVBRDF.

Figure 2.11: Comparison of image analogies approaches for input shown on the left, of the *denim* material. From left to right: Deep Image Analogies [Lia+17] and Structural Analogies [Ben+21]. Our results shows more accurate and predictable mappings, at less computational cost than the alternatives.

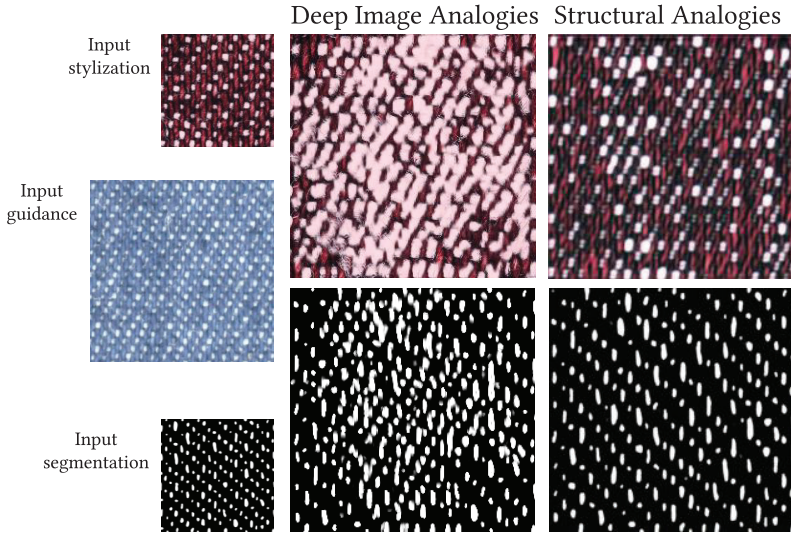


Figure 2.12: Our method allows for interactive material-aware visual attribute transfers. Using an off-the-shelf style transfer algorithm [GEB15b], we can transfer the style of one image I_{style} to the content of another $I_{content}$, obtaining a visual attribute ω . Training a M_ω to learn this relationship, we can find predictable style mappings, that we can transfer to guidance images X , obtaining style transfers $M_\omega(X)$. Learning this transfer is inexpensive and allows for interactive editions. Performing this transfer directly to the guidance image generates artifacts and not-predictable mappings.

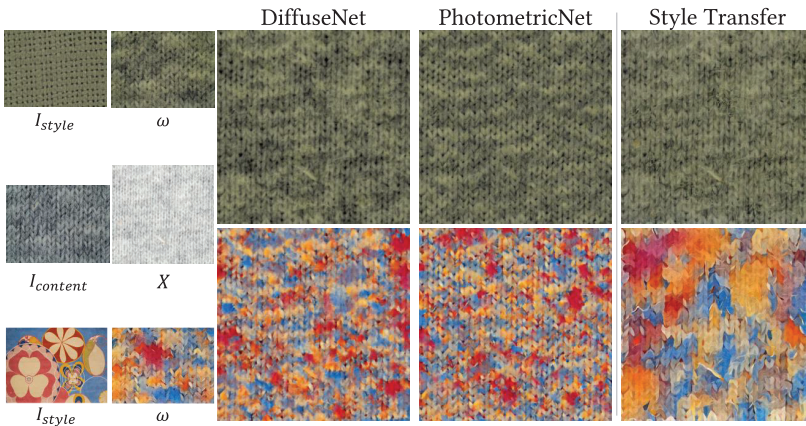


Figure 2.13: Results of our framework for material capture using a smartphone. Training two models, M_a and M_n with a photometric dataset, we obtain respectively albedo and normals from guidance images taken under uncontrolled conditions, which can be used by render engines. Here we have used Arnold [Geo+18] and a diffuse material model.

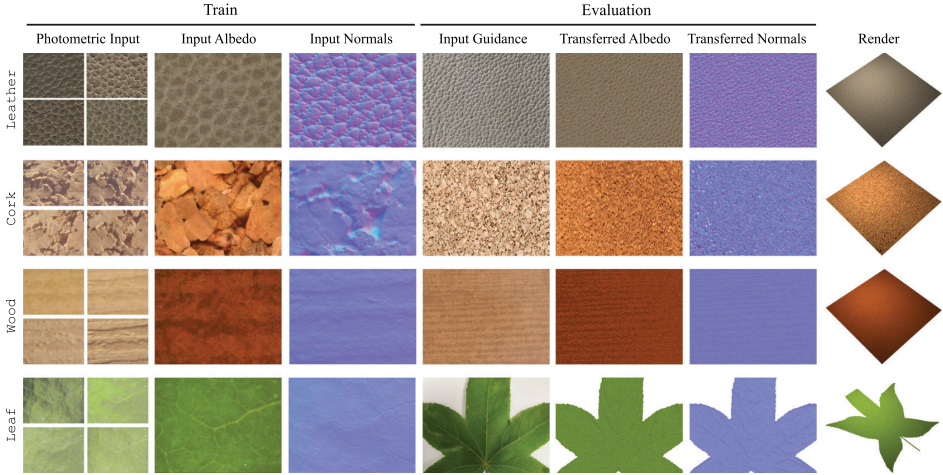
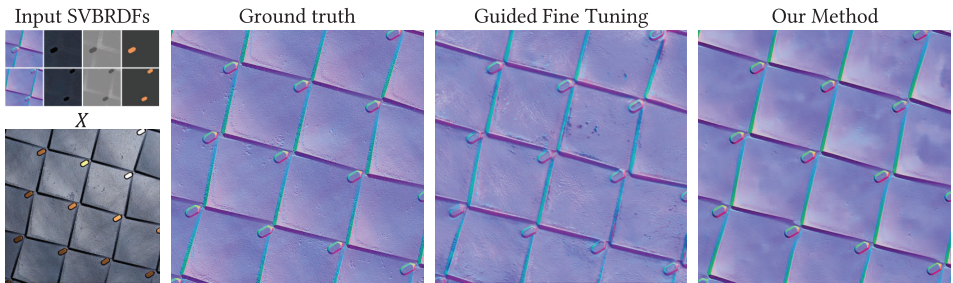
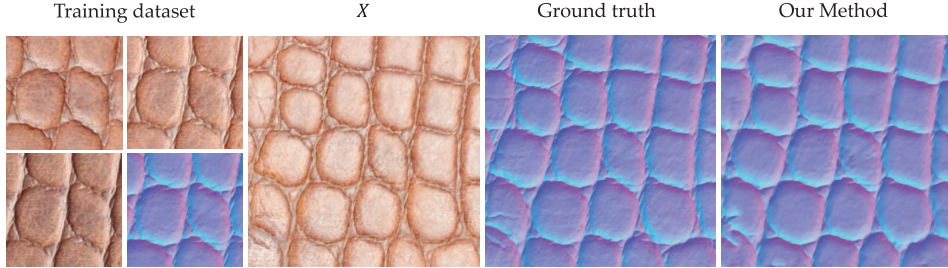


Figure 2.14: Comparison of our method with the Guided Fine Tuning, by Deschaintre *et al.* [DDB20]. Following their algorithm, we render the *Input SVBRDFs*, and train a photometricNet on those renders. As shown, our method can achieve higher quality normal maps, with fewer artifacts. Input SVBRDFs, X , ground truth and results from Guided Fine Tuning were obtained directly from [DDB20].



We assess the capabilities of our method on this setting using the same SVBRDF propagation scenario as proposed in [DDB20]. Using a small crop of a synthetic SVBRDF as input, we render 27 images using the same directional light position as we used in our real dataset, and train a photometricNet to estimate the surface normals from each of those renders. We then evaluate this model using a larger area of the material, illuminated under an unknown lighting position. In Table 2.1, we show a quantitative comparison

Figure 2.15: Our method can work with real BTF captured data. Training a photometricNet on a crop of the material (summarized in *Training Dataset*), to output surface normals, we can estimate surface normals on larger areas of the material, even under novel viewing positions.



with [DDB20], under different image quality metrics. As shown, our method achieves better scores on pixel-wise metrics, whilst [DDB20] achieves better deep perceptual scores, as in the LPIPS metric [Zha+18]. This might be related to the design of our loss function: we directly minimize pixel-wise differences, while [DDB20] is optimized using a render-aware loss. Qualitatively, as shown in Figure 2.14, our method obtains comparable quality mappings, with fewer artifacts. Despite capturing SVBRDF or arbitrary materials is not the goal of our method, we include in the supplementary comparisons with a direct SVBRDF acquisition method [Gao+19] using our own real dataset.

Table 2.1: Quantitative comparison with [DDB20], on the studied materials and different performance metrics. As shown, our method provides better pixel-wise accuracy than [DDB20], while their method obtains better perceptual scores.

Material ID	SSIM [Wan+04] $\uparrow$		PSNR $\uparrow$		MSE $\downarrow$		LPIPS [Zha+ 18] $\downarrow$	
	[DDB20]	Ours	[DDB20]	Ours	[DDB20]	Ours	[DDB20]	Ours
560	0,905	0,925	31,190	32,310	0,002	0,002	0,244	0,221
1581	0,645	0,675	29,740	29,920	0,006	0,005	0,398	0,446
1684	0,654	0,746	29,711	30,410	0,007	0,003	0,196	0,251
2111	0,770	0,783	32,250	32,881	0,002	0,002	0,397	0,383
Average	0,744	0,782	30,723	31,380	0,004	0,003	0,309	0,325

A similar setting can be used to propagate attributes leveraging BTF measurements as training data. Figure 2.15 shows an example using a BTF from [WGK14]. Using Photometric Stereo [Ike81], we compute their surface normals and train a photometricNet on a central crop of the BTF, using the captures at a camera position of ($\phi = 0^\circ$, $\theta = 0^\circ$). We then evaluate this model using the full

material surface, and a novel camera position, of ($\phi = 15^\circ$, $\theta = 11^\circ$). As shown, our model is capable of working with captured BTF data.

2.7.3 Generalization to Similar Materials

In previous experiments, we have shown the performance of our models when the guidance image corresponds to the material used for training. In this experiment, we show that our models, despite being trained only on a set of captures of a single material, generalize to materials of the same category. Figure 2.16 shows some examples for models trained using our dataset (Figure 2.5), taking input guidance images of different albedos and scales. The transfer works thanks to the network design and data augmentation strategy that is designed to use microstructure details as guidance. Our method could thus be used to transfer visual attributes for a diverse set of materials by simply training one model using a single but representative material of each category.

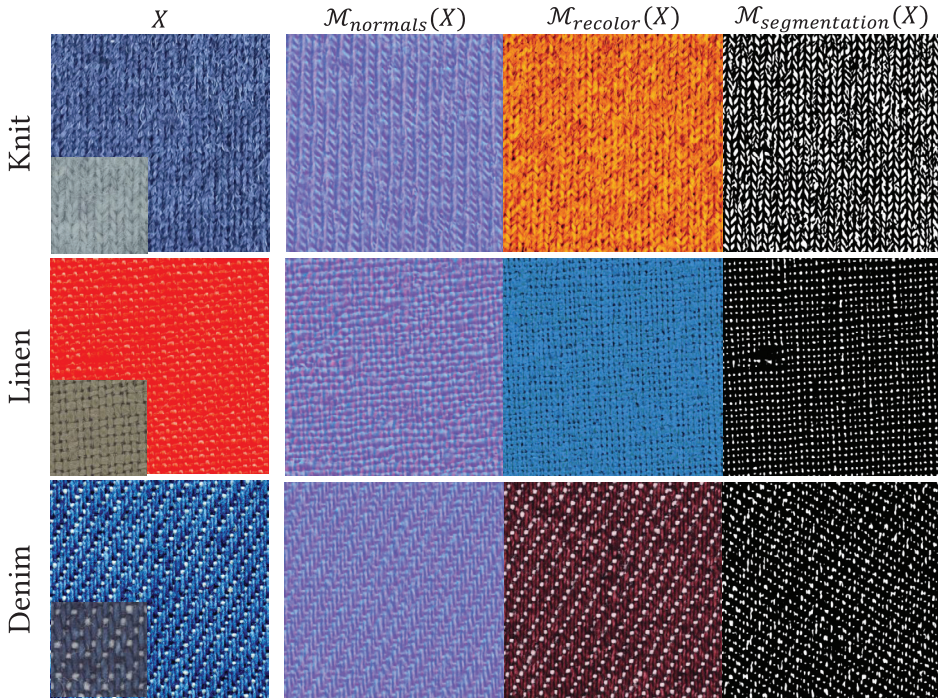
2.7.4 Limitations

Our models are not guaranteed to provide high-quality results outside the range of input data and data augmentation policies we train them on. This limitation is common to all learning-based approaches. It is unlikely that the framework is capable of generalizing to resolutions higher to those of the training data; or down-sampled images in which the texture details are not recognizable. Furthermore, the type of transformations we apply to the training data may not represent all the possible geometric variations, or non-linear warpings that materials are subject to in the real world. For materials which exhibit a strong variation in their microstructure, which cannot be fully captured using a single photometric dataset, our patch-based approach will likely fail to generalize to the full heterogeneity. Figure 2.17 shows two examples of failure cases for which the test data is not included in the training data and where the affine and illumination transformations are outside the suitable range.

2.8 Conclusions and Future Work

In this chapter, we have proposed a neural visual attribute transfer framework capable of transferring, for a given material, many types of visual property maps to images of unseen patches of the same -or similar- material taken under different illumination, capture setup, and affine distortion. To our knowledge, the proposed framework is the first method capable of leveraging the optical behavior of the material to this purpose by being trained using a photometric approach. Such an approach, besides the illumination-invariance we have shown, helps

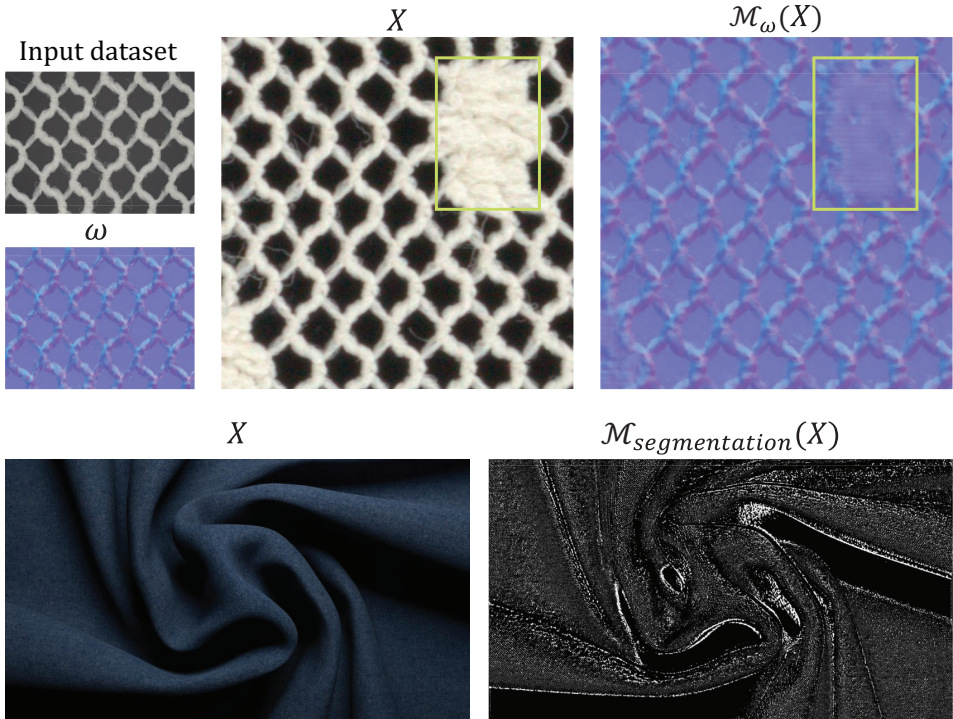
Figure 2.16: Generalization capabilities of our method when evaluated on materials similar to those in their training dataset. On the top row, we show the outputs of the model trained on the *knit* in our dataset (Figure 2.5), and evaluated on a different guidance images. We also show examples on linen and denim fabrics, with different conditions of saturation, blur, scale, and illumination. Even in very challenging cases, where the structure of the material is barely visible, as in the overly-saturated red linen or the noisy blue knit fabrics, our photometricNets can yield plausible results. The insets represent the training dataset by each model, as represented in Figure 2.5.



the neural network learn better mappings between visual domains, finding a physically-based representation of the material. Further, we have presented a comprehensive policy of data augmentation which outperforms previous work on visual attribute transfer given a single image of the material.

We have shown that our method can be used to transfer any kind of visual attribute estimated locally to larger material samples. Further, we have demonstrated that our models, although trained on a single material, generalize to materials of the same category. We think our findings will inspire future work showing that smart training strategies might alleviate the need for massive datasets. Our method could be extended in several ways. The need for obtaining high-resolution captures taken under

Figure 2.17: Failure cases of our method. As shown in the first row, if the input dataset does not represent the heterogeneity present in the guidance X , the model fails to yield compelling results on unseen structures of the material, as shown on the green box. On the second row, we show a guidance image of the *denim* material X which exhibits strong geometric and illumination variations, outside of the range in which we train M with. As such, the model shows a poor performance on the segmentation task.



different illuminations may be reduced by generating rendering images through recent advances in inverse material acquisition [Don19]. Further training with multiple patches may help to cope with material heterogeneity [Tex+20b]. Similarly, our findings suggest that extending the data augmentation policy to include 3D deformations will likely improve the accuracy. Beyond the generation of large scale digital assets for rendering, our method may have potential in other visual computing applications that require a low level understanding of the properties of the materials in real scenes. For example, the yarn segmentation application shown in the chapter might be suitable as input to shape from texture applications. Specific visual attributes might be useful to identify or highlight defects for image forensics problems, or to enhance different features in real-time or AR applications.

2.A Additional Implementation Details

Network design

Our model, summarized in Figure 2.18, is a standard U-Net [RFB15] encoder-decoder network with 4 blocks of layers and skip connections. U-Nets are widely used in pixel-wise regression and image-to-image translation problems, due to their efficient design, capable of exploiting texture and semantic patterns at different levels of abstraction, due to their multi-scale design. Skip connections are known to significantly enhance the quality of deep image-to-image translation models [Iso+17]. Each block is comprised of a tuple of convolutional layers with kernel sizes of 3×3 , followed by a batch normalization operation [IS15] and a ReLU [NH10] non-linearity. The number of convolutional filters in each layer is shown in Figure 2.18. The last layer of the network uses 1×1 convolutional filters to transform the feature vectors into the desired number of classes (e.g 3 output channels for RGB regression tasks, as in normal map estimation or recoloring; or 1 output channel for the semantic segmentation problem). The total number of trainable parameters depends on the number of property maps that the network should jointly learn, but are around 483000 for all the problems we showcased in this work. Trainable weights are initialized randomly, by sampling a $N(\mu = 0, \sigma = 0.02)$. The output of each block in the encoder part of the network is max-pooled to halve its spatial resolution. Upsampling in the decoder side of the network is done through up-convolutions. For image segmentation problems, we add a Sigmoid non-linearity to the output of the network. Due to the reduced number of trainable parameters, models take only 1.94Mbs, allowing for fast training and evaluation. On Table 2.2, we show a quantitative study on the influence of the depth of the model in its accuracy, using the 1684 material from [Des+18] as the training dataset. A visualization of these results is included in Figure 2.19, showing that deeper models yield overly-smooth results for this few-shot learning problem.

Table 2.2: Quantitative comparison between different number of blocks of layers in our model. Every model we use throughout this work has 4 blocks of layers, as it provides the best trade-off between computational cost and precision. Deeper models fail to generalize properly in this few-shot learning scenario. We use a color code to highlight **best** and **worst** cases

Depth	SSIM [Wan+04] $\uparrow$	PSNR $\uparrow$	MSE $\downarrow$	LPIPS [Zha+18] $\downarrow$	Size (Mbs) $\downarrow$	Train Time (s) $\downarrow$
3	0.71	29.83	0.0052	0.37	0.49	51
4	0.74	30.41	0.0031	0.25	1.94	58
5	0.72	30.37	0.0043	0.33	7.63	164
6	0.72	30.87	0.0041	0.31	30.49	283

Figure 2.18: An overview of our deep learning architecture M_{ω_l} , which is a shallow U-net network [RFB15]. Below each block of layers, we show the number of trainable convolutional kernels, as well as the spatial resolution spanned by each block, with respect to the input resolution D . We use the normal map estimation problem for a denim fabric for visualization purposes.

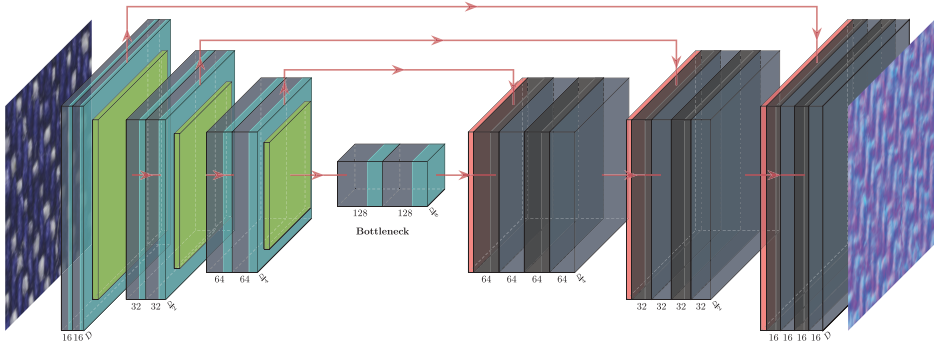
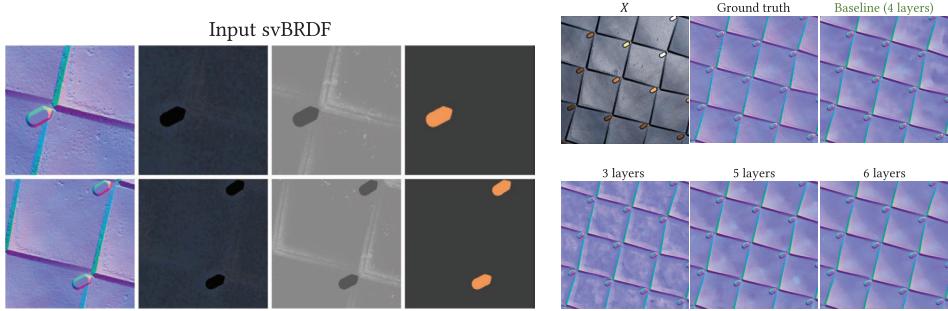


Figure 2.19: Impact of the depth of the model on its accuracy, using a synthetic dataset. As shown, our baseline outperforms deeper and shallower configurations.



Training details

We use PyTorch [Pas+19] as the learning framework, Adam [KB15] for optimization, a learning rate of $\alpha = 0.002$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and a batch size of 16. The training images are randomly augmented using a uniform distribution by the following operations, in order: randomly rotated by an angle in the $[-90, 90]$ range, randomly sheared by an angle in the $[-45, 45]$ range, and randomly rescaled in the $[0.5, 2]$ range of scale factors. Their color channels are randomly reordered and patches of 128×128 pixels are randomly cropped during training, to generate a large dataset of images. The rescaling factors and rotation angles are randomly chosen for each image in each batch using a uniform distribution on the mentioned ranges. All these data augmentation

operations are also applied to the target visual attributes, except for the color channel reordering, which is only applied to the input images. Images and visual attributes are always standardized using their own mean and variance. All the data augmentation operations are performed natively in GPU, which allows for faster training. To further reduce the computational cost and training times of our models, we train our models using automatic mixed precision training [Mic+18], which reduces the memory footprint of the models in GPU, while allowing for faster computation of certain operations, such as convolutions.

Each M_ω is trained independently for each material and application (no cross-domain training or fine-tuning) for 1000 iterations, which takes around 1 minute on a single 1080Ti GPU. Due to the fully convolutional nature of our models, and their reduced number of trainable parameters, the guidance images X used for evaluation can be of arbitrary resolutions. We evaluate the models using half-precision, which allows us to utilize guidance images of up to 5000×5000 pixels, for which evaluation takes around 150ms in GPU. Thanks to the reduced number of parameters, an efficient use of GPU-native data augmentation techniques and leveraging half precision training and evaluation, we are able to train and evaluate our photometricNets in less than a minute of total computational time.

Ablation Study of the Loss Function

Finally, in Figure 2.20, we show a comparison of the results of our models when trained on different loss functions. As shown, even though the differences are sometimes subtle, the L_1 consistently yields sharper estimations than the L_2 , whilst the perceptual loss function is more artifact-prone and thus less predictable. L_1 is a common loss function in many image regression tasks, as in image-to-image translation [Iso+17] or in image synthesis [Zho+18].

Additional Dataset Details

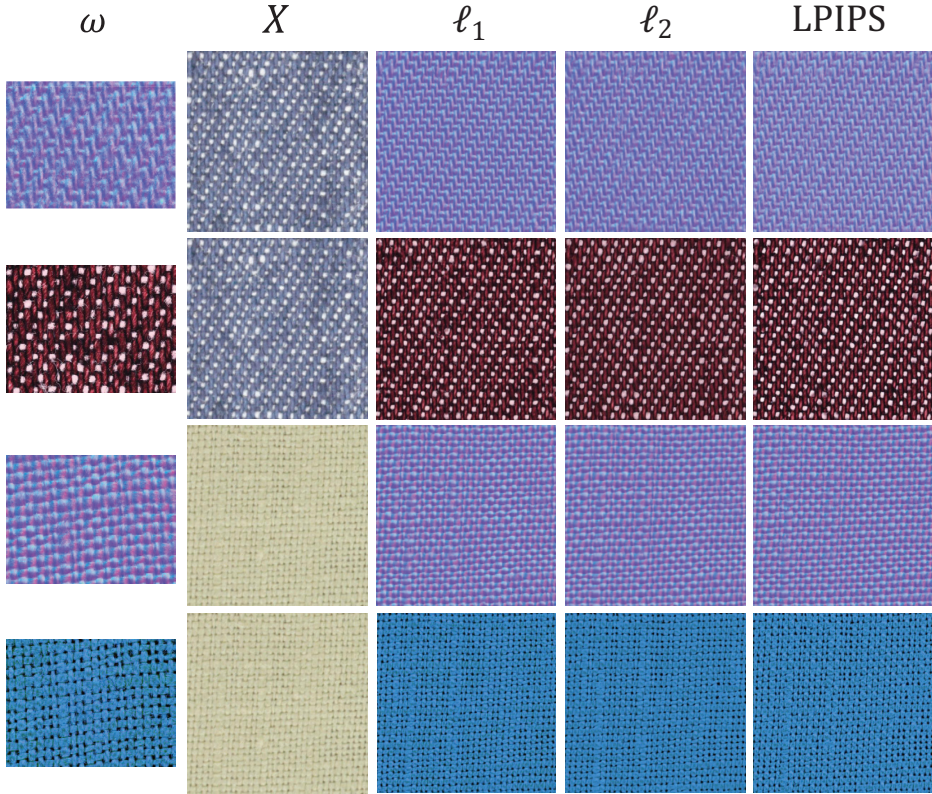
We show the positions of the lights in the photometric dataset in Figure 2.21.

2.B Additional Comparisons

Comparison with Image Analogies Methods

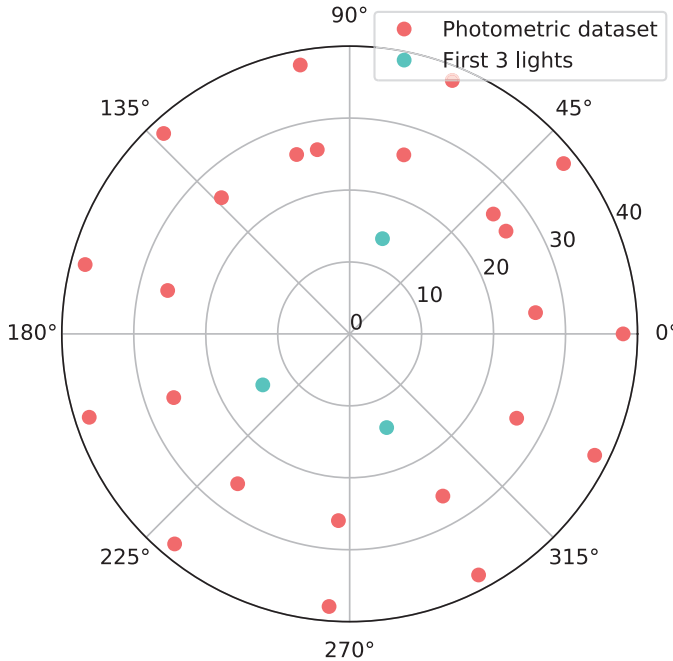
In Figure 2.22, we compare our method to different image analogies approaches. We show a comparison between [Her+01], [Lia+17], [Ben+21], and our method. The seminal work in image analogies [Her+01] fails in this visual attribute transfer problem, most likely due to the difference in scale between input and target images. For the comparisons with [Lia+17] and [Ben+21], we reduced

Figure 2.20: Impact of the choice of the loss function used to train the networks in the estimated maps at macroscale. On the left, input guidance X and visual attribute ω to be transferred. On their right, the result of M_ω when trained using L_p , L_2 and $LPIPS$ [Zha+18] loss functions. All the networks in this experiment were trained with the same configuration, only the loss function was modified.



the scale of the target images B to 256×256 pixels to make their methods computationally tractable. The methods in [Lia+17; Ben+21] never see the diffuse image A , as they generate an intermediate representation by themselves. While showing better results than [Her+01], methods which use deep learning for this task still fail to yield compelling or predictable results. The method in [Ben+21], which leverages structural information in the images to train a single-image generative model [SDM19], shows more plausible results, but with an expensive computational footprint (10 hours per network), compared to the 45 minutes in [Lia+17], 10 minutes in [Her+01] and less than 1 minute in our case. Because we assume pixel-wise semantic correspondence between input image and visual attributes, we can find mappings in interactive times, and which are significantly more predictable and of higher quality than these image analogies methods.

Figure 2.21: Relative position of the 27 directional light sources used to capture the photometric datasets with respect to the material sample, in degrees. In blue, we show the three lights that are always included in our reduced photometric datasets for the light sampling experiments. As shown, the lights cover a wide variety of azimuth and zenith angles, allowing for an accurate capture of the optical behavior of the materials.



Comparison with Texler et al.

In Figure 2.23, we show a comparison with different configurations of [Tex+20b]. Specifically, we train their method using a dataset with a single image using diffuse illumination (left) and the full photometric dataset (middle column). Their method is trained individually for each dataset and visual attribute, for around 5 minutes to guarantee convergence, as suggested in their paper. As [Tex+20b] assumes that the scale (pixels/cm) of the dataset and evaluation images are the same, we rescale our input dataset to match the scale of the evaluation images. We evaluate each method using three different images X , one with diffuse-light illumination (top row), another with diffuse-light illumination but with a geometric transformation (middle row), and an image captured with directional lighting (bottom row). As shown, by enhancing the method in [Tex+20b] with a photometric dataset, their model is able to generalize better to every image X in the evaluation dataset than its counterpart trained only using a single image. However, our method, specifically

designed for this illumination and geometry invariant visual attribute transfer outperforms their work in every type of input and visual attribute, whilst maintaining a lower computational overhead (1 minute of training in our case) and memory footprint (their trained network uses up 12 Mbs, while ours is 1.9 Mbs). Our model trained using a single illumination (diffuseNet) shows better results than their method trained using the photometric dataset, suggesting that our transfer method has more adequate inductive biases for the visual attribute transfer problem.

Comparison with deep SVBRDF acquisition

In Figure 2.24, we show a comparison with a recent deep SVBRDF acquisition method [Gao+19], which relies on a pre-trained autoencoder and a differentiable render engine for optimizing the initial estimation of the autoencoder. As shown, our method generates sharper and more coherent normal maps. PhotometricNet relies on a small sample of the same material for training, and is thus less sensitive to potential biases of the training dataset, with the cost of having reduced generalization capabilities compared to multi-material methods.

Figure 2.22: Comparison of photometricNet with related image analogies methods.

In order, we show a comparison between [Her+01], [Lia+17], [Ben+21], and our method. Our method clearly outperforms any analogies method, at a fraction of the computational cost. The comparisons with [Her+01] were done using an unofficial implementation of their method.

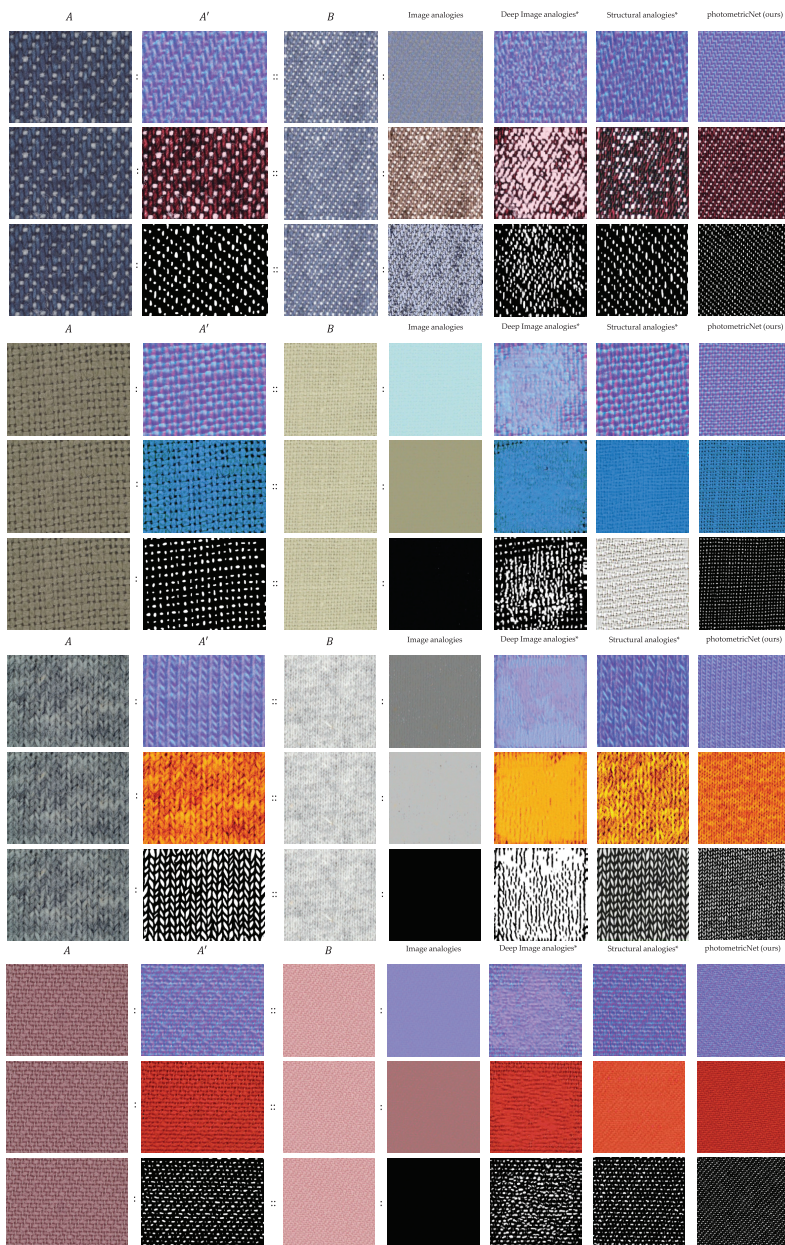


Figure 2.23: Comparison of our method with different configurations of [Tex+20b]. On the top row, we compare their results and ours using a dataset containing only one image, captured with diffuse lighting. On the bottom row, we show results when training these methods with a full photometric dataset. Our method is able to generalize better to different illumination and geometric conditions, with a smaller computational overhead.

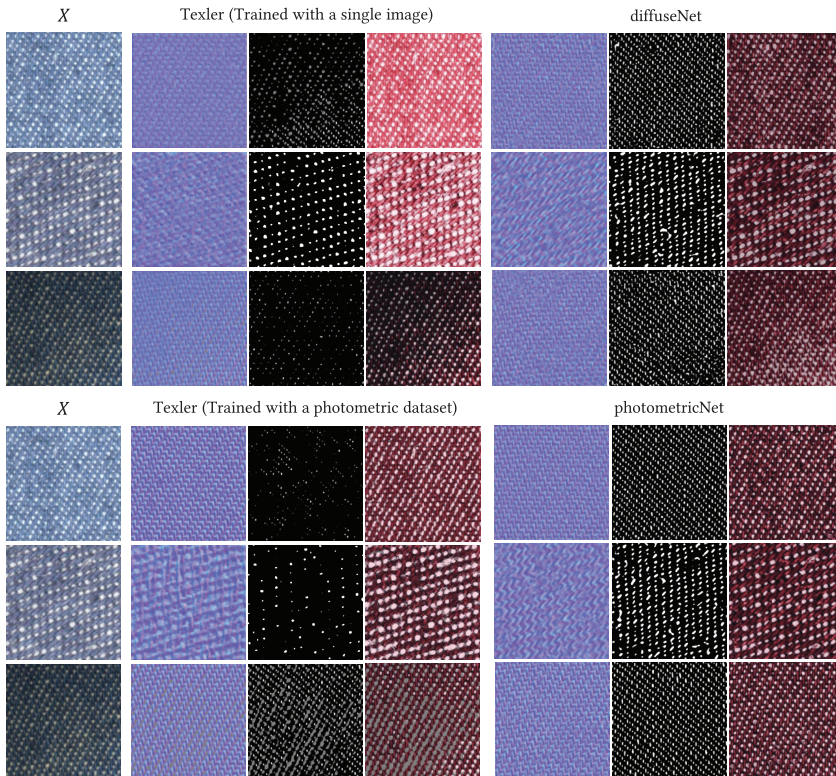
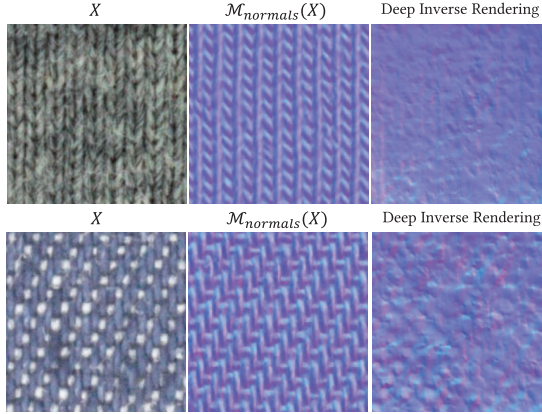


Figure 2.24: Comparison of our method with [Gao+19]. From left to right, we show the input guidance X , the output of photometricNet $\mathcal{M}_\omega$ trained to transfer normal maps, and the output of [Gao+19] after 5000 optimization steps. Their method relies on a pre-trained network, which learns a prior over materials that does not generalize to examples outside their training set. The images in this experiment are of 256×256 pixels, to match the input requirements in [Gao+19].



2.C Extension to SVBSDF Propagation

Figure 2.25: Renders of SVBSDF maps transferred by our extension of photometricNet.



In this section, we present an extension to the previous method for propagating material maps. With it, we can propagate multiple property maps at the same time, learn from many photometric datasets simultaneously, and achieve more accurate

and sharper results. With these improvements, we are able to propagate multiple SVBSDF maps at the same time, allowing for efficient high-resolution material generation, as we show in Figures 2.25 and 2.26. These findings are part of the data generation process which is used to build the datasets we use to train the models described in Section 5 and [Rod+23a], and are also part of a paper on high resolution material digitization, which has been accepted to *SIGGRAPH 2023*:



"Towards Material Digitization with a Dual-scale Optical System" Elena Garces, Victor Arellano, Carlos Rodriguez-Pardo, David Pascual-Hernandez, Sergio Suja, Jorge Lopez-Moreno

ACM Transactions on Graphics (Proceedings of SIGGRAPH) (2023)

2.C.1 Mesoscale Propagation

Figure 2.26: Left, virtual scene rendered using eight different materials digitized with our optical device. Right, real photos of four of the materials taken under diverse illumination conditions: area lights, diffuse lighting, and directional lighting of a high-resolution patch.



To propagate material properties, a common solution is to utilize the estimation of material properties in a localized area and extended it to larger samples [ND06; RSK13; Ste+14; DDB20]. We follow this approach to propagate SVBSDF measurements, and build upon the work presented in Section 2 and [RG21],

which leverage images of the material taken under different illumination conditions to train a neural network that propagates individual maps.

As input to this propagation step, we use a photometric dataset obtained with a microscopic setup, spatially-varying parameters of a micro SVBSDF (Illustrated in Figure 2.28), and a guidance image that will serve as a reference for the mesoscale propagation. We use as guidance the image taken with a mid-range camera and diffuse lighting. Unlike the previous approach, aimed at transferring individual maps, we introduce some modifications to jointly propagate the full stack of parameters. We further adapt the method to be able to take as input more than one micro SVBSDF, necessary if the variability of material is not visible in a single micro capture (see Figure 2.27 (a)). We propose the following modifications:

- 1. Larger Training Dataset.** We improve the generalization of the model by using more training data in two ways: First, we use a larger photometric dataset, using more directional-lit images than what is described in 2.21, as well as diffuse lighting. Second, we use random Gaussian blurs for data augmentation to improve robustness to potential degradation in mesoscale guidance images, and remove affine distortions to the data augmentation policy as they are not required for our SVBSDF transfer problem. Additionally, we train the model for more iterations, with a larger batch size.
- 2. Enable Large Spatially-varying Materials.** We account for materials in which the appearance variability is not observed in a single micro capture by training the propagation network with as many micro SVBSDF as necessary. Figure 2.27 (a) showcases an example that required two microscale captures. Further, we disable the color-invariance data augmentation that randomly shuffles color channels when multiple micro SVBSDFs are available.
- 3. Improved Architecture and Losses.** Finally, we observed that the original architecture was not able to effectively transfer a large number of parameters at the same time. Therefore, inspired by recent work on intrinsic decomposition [Jan+17], texture synthesis [RG22], and material estimation [DLG21], we use a U-Net [RFB15] with a single encoder but a separate decoder for every map we aim to transfer. Further, we introduce residual connections [He+16; Dia+20] into every layer, and substitute Batch Normalization [IS15] for Group Normalization [WH18]. When multiple microscale captures are available, we use a model with additional filters in every layer, to better learn from these extended datasets. This design generates sharper texture maps which preserve the statistics and appearance of each fitted microscale map more accurately. We also added a multi-channel perceptual component [CHB21] to the loss function, which is a powerful

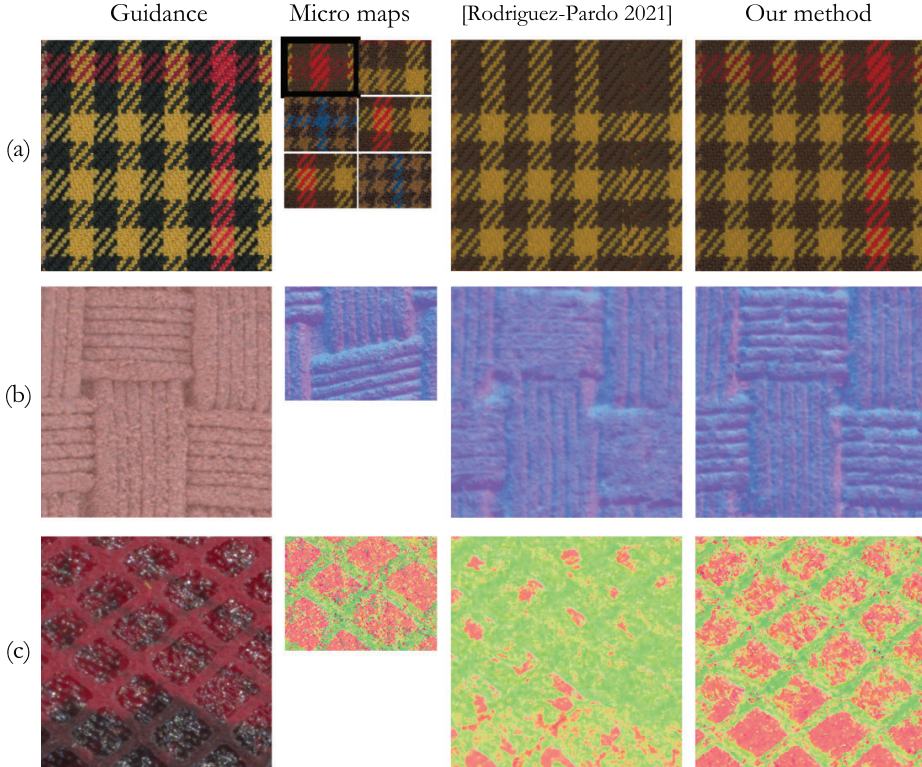
regularizer for texture synthesis. Figure 2.27 (b-c) shows the difference with respect to the previous approach.

2.C.2 Implementation Details

Problem Formulation

As in [RG21], our goal is to transfer spatially-varying attributes of a material captured at microscale to larger samples of the same material. We thus formulate this problem using an image-to-image translation framework. For training, our methods take as input a *set of D photometric datasets* of a material $\mathcal{I}_L^D$, and their pixel-wise corresponding SVBSDFs M_p^D , each comprised of a list of P maps: $p \in \{\text{albedo, roughness, transmittance, IOR, anisotropy, tangents, normals, specularTint, opacity}\}$. Each photometric dataset is comprised of a set of RGB images of the material captured under different illumination conditions: $\mathcal{I}_L^d = \{i_l^d \mid i_l^d \in \mathbb{R}^{n \times m \times 3}\}, |\mathcal{I}_L^d| \geq 1, \forall d \in D$. In contrast to [RG21], which only allowed to learn from a single photometric dataset and to transfer a single attribute, our extended approach allows for learning from multiple datasets $|D| \geq 1$ and multiple property maps $|P| \geq 1$ using a single model. We train a model T which learns to transfer from each image in each photometric dataset to its corresponding SVBSDF: $T(i_l^d) \approx M_p^d, \forall d \in D, l \in L$. During inference, we use a $1 \times p$ guidance image of the material X , which represents a larger sample of it, and estimate its corresponding property maps: $M^X \leftarrow T(X)$. At evaluation time, this guidance image X may be captured with different conditions to those in the training dataset, including different camera or illumination conditions.

Figure 2.27: Comparison with photometricNet [RG21] for mesoscale maps propagation. The first row (a) showcases an example where multiple captures at microscale were needed to cover the spatially-varying albedo of the material. (b) and (c) required a single capture, we show the result of normals and tangents maps compared with previous work.

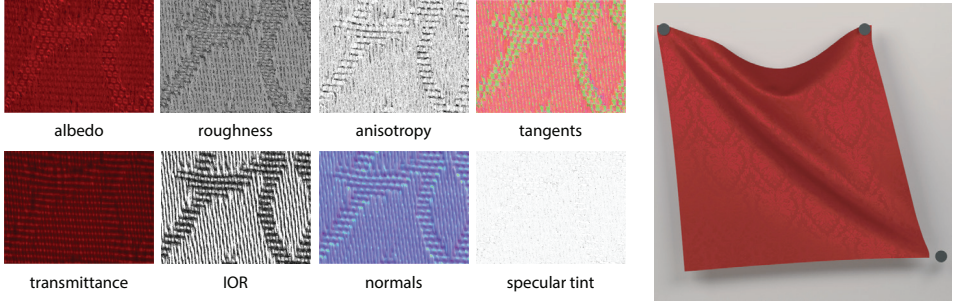


Datasets

We use a more comprehensive photometric dataset to what is proposed in [RG21]. In particular, for a single microscale capture, we use all the light sources used in the fitting algorithm, as well as diffuse-lit microscale images. Additionally, we use both polarization modes separately and, for the same light source, we construct an extra image by averaging the images taken under both polarization configurations. The total size of each photometric dataset is of $|\mathcal{I}_L^d| = 108, \forall d \in D$, four times more data compared with the more limited 27 images proposed in [RG21]. For evaluation, our guidance image is taken with a mid-range camera and *diffuse* illumination, at a resolution of 4072×4072 pixels and a surface area of 10×10 centimeters. We typically use a single photometric dataset ($|D| = 1$). However, for multi-colored or

highly heterogeneous materials in which a single capture is not enough to fully represent the material variability, we capture as many datasets as needed.

Figure 2.28: On the left, microscale SVBSDF maps. On the right, a render generated with the maps propagated with our method.



Loss Function

Our loss function is the combination of two terms: weighted pixel-wise losses for each target map, and a multi-map style loss:

$$\mathcal{L} = \sum_p \lambda_p \mathcal{L}_{pixel_p} + \lambda_{style} \mathcal{L}_{style} \quad (2.3)$$

$\mathcal{L}_{pixel}$ is the L_1 norm weighted per map in the SVBSDF, $\lambda_p \cdot L_1$ produces sharper results than higher-order alternatives [RG21], such as L_2 . For the $\mathcal{L}_{style}$ loss term, we follow recent work on multi-channel texture synthesis, which extends deep perceptual losses for style transfer to the SVBRDF synthesis problem [CHB21]. This component of the loss function acts as a regularizer and allows the model to generate maps that better preserve the appearance of the ground truth training data.

Training Implementation Details

Model Design We use a lightweight U-Net [RFB15] architecture, with a few modifications based on recent work to maximize its efficiency and the quality of its outputs. We specify the full model architecture and model sizes in Figure 2.29. In every convolutional block of the model, we use residual connections [He+16; Dia+20], for better training convergence and preservation of details in the input images. We use 1×1 convolutions on these residual connections. We use a single decoder for each map, so as to maximally preserve their individual appearance and statistics. This has been proposed in recent work on texture synthesis [RG22], material capture [DLG21] or intrinsic decomposition [Jan+17]. On the convolutional blocks, we leverage Group Normalization [WH18]. Upsampling is done using transposed convolutions. As shown in Figure 2.29, the model has four hidden layers on the encoder and each decoder, however, we vary the size of those layers

depending on whether the model is trained with a single microscale capture (for which we use a width factor of $W = 16$) or multiple ($W = 32$). Every other implementation detail in the model (stride, bias, pooling) follows [RFB15; RG21].

Training and Evaluation We use PyTorch [Pas+19], Torchvision [MR10], and Kornia [Rib+20] for training. We leverage mixed precision training and automatic gradient scaling [Mic+18], to accelerate the training process and regularize the models. Optimization is done using Adam [KB15] for 5000 iterations, with a learning rate of 0.002, a batch size of 40, and no weight decay. This process takes around 15 minutes when training with a single microscale capture, and 25 when using multiple captures as input. We evaluate the guidances images using half precision. We measure these times on an NVIDIA 1080Ti GPU.

Data Augmentation We follow the same training procedure defined in [RG21]. However, we do not use random rotations or shears, and remove the color invariance data augmentation whenever multiple microscale captures are available. Further, we introduce random Gaussian Blurs for data augmentation, using a $p = 0.5$, a kernel size of 5, and sigma selected uniformly at random for each element in each batch: $\sigma \sim U(0.1, 11)$. As in [RG21], we also perform random cropping, using crops of 128×128 pixels. For each element in each batch during training, we randomly choose the photometric dataset $d \in D$, and, from it, a random light source $l \in L$.

Loss Function We weight the loss function as follows: For the pixel wise loss: $\lambda_{\text{normals}} = 3, \lambda_{\text{IOR}} = 1, \lambda_{\text{roughness}} = 1, \lambda_{\text{albedo}} = 3, \lambda_{\text{tangent}} = 3, \lambda_{\text{anisotropy}} = 1, \lambda_{\text{specular}} = 1, \lambda_{\text{Tint}} = 1, \lambda_{\text{transmittance}} = 3, \lambda_{\text{opacity}} = 1$. The perceptual component [CHB21] is weighted with $\lambda_{\text{style}} = 0.25$, and we use the *AlexNet* variant of *LPIPS* [Zha+18] as our backbone, which provides a powerful yet lightweight loss for texture transfer, as shown in [RG21].

2.C.3 Results

In Figure 2.30, we show an ablation study on the improvements of the maps propagation method proposed in [RG21]. We build upon their proposed implementation, removing their random shifts and rotations, and make progressive changes, in order, to the dataset size, model architecture, loss function, and data augmentation. We observe high-quality propagations, but the baseline sometimes produces overly smooth outputs (Fleece, Leather-Brown, Rib-Silver, or Leather-Lizard). Our proposed increased dataset and model architecture modifications enhance the maps' sharpness, and our perceptual loss function allows for the propagation of additional details. Our final model removes the color augmentation proposed in [RG21] whenever multiple microscale captures are available, and introduces random blurs during training, allowing for achieving the highest quality results and for accurate base color propagations in challenging cases, as in Tartan.

Figure 2.29: A full diagram of the model architecture we use for our mesoscale propagation problem. Building upon [RG21], use a lightweight U-Net [RFB15] architecture. Following previous work [Jan+17; RG22; DLG21], we use a different decoder for every map we aim to transfer. Each of our convolutional blocks, illustrated on the right, contain residual connections [He+16; Dia+20] and Group Normalization [WH18]. In red, we show the input/output dimensions (spatial, channels) of each layer; in blue, convolutional blocks and layers; in green, upsampling and concatenating operations; in yellow, normalization layers; and in purple, regularizations, and non-linearities. We set W to $W = 16$ when a single microscale capture is available as training data, $W = 32$ otherwise.

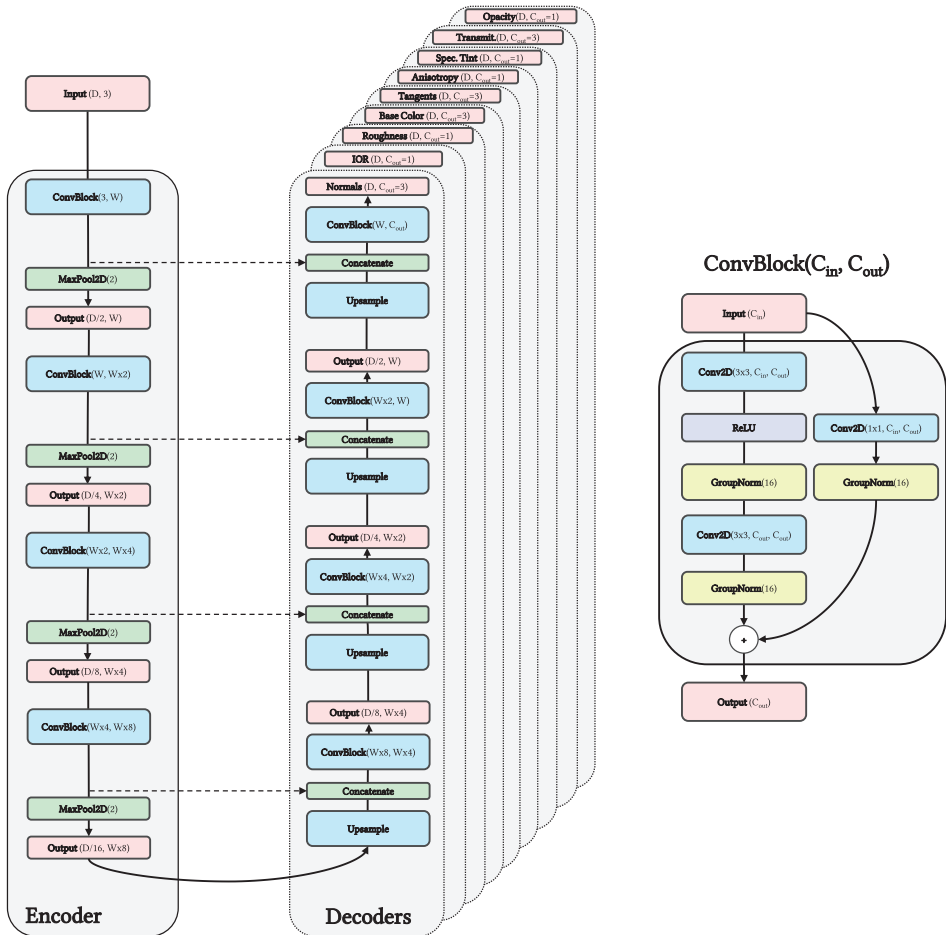


Figure 2.30: Ablation study of our proposed improvements with respect to tphotometricNet [RG21], on base color (top rows), normals (mid) and tangent map (bottom) transfer. We start from the implementation in [RG21], which shows adequate but smooth results. On its right, we show the impact of increasing the dataset size, which tends to generate higher-quality estimations (Leather-Brown).

With our improved architecture (fifth column), we achieve sharper maps (see Leather-Lizard, Rib-Silver, or Fleece). Using a perceptual loss (sixth column), we further improve the maps quality (Leather-Brown). Our final model (last column), which removes the color augmentation in [RG21] and introduces random blurs to the model training, achieves the highest quality, and allows for accurate base color transfers, as shown in Tartan. Input guidance images and training micro maps are shown on the first and second columns, respectively.

