Interpretable Disentangled Parametrization of Measured BRDF with $\beta$-VAE

Alexis Benamira, Sachin Shah, Sumanta Pattanaik
University of Central Florida

Abstract
Finding a low dimensional parametric representation of measured BRDF remains challenging. Currently available solutions are either not interpretable, or rely on limited analytical solutions, or require expensive test subject based investigations. In this work, we strive to establish a parametrization space that affords the data-driven representation variance of measured BRDF models while still offering the artistic control of parametric analytical BRDFs. We present a machine learning approach that generates an interpretable disentangled parameter space. A disentangled representation is one in which each parameter is responsible for a unique generative factor and is insensitive to the ones encoded by the other parameters. To that end, we resort to a $\beta$-Variational AutoEncoder ($\beta$-VAE), a specific architecture of Deep Neural Network (DNN). After training our network, we analyze the parametrization space and interpret the learned generative factors utilizing our visual perception. It should be noted that perceptual analysis is called upon downstream of the system for interpretation purposes compared to most other existing methods where it is used upfront to elaborate the parametrization. In addition to that, we do not need a test subject investigation. A novel feature of our interpretable disentangled parametrization is the post-processing capability to incorporate new parameters along with the learned ones, thus expanding the richness of producible appearances. Furthermore, our solution allows more flexible and controllable material editing possibilities than manifold exploration. Finally, we provide a rendering interface, for real-time material editing and interpolation based on the presented new parametrization system.

CCS Concepts
• Computing methodologies → Reflectance modeling; Learning latent representations;

1. Introduction
Natural material appearance is rich and varies over a large set of dimensions. This wide range of possible variations makes material appearance a hard function to represent and developing a corresponding parameter space is just as challenging. A function that is commonly used to represent material appearance is the Bidirectional Reflection Distribution Function (BRDF) and it falls into two main categories: analytical or measured. Analytical BRDFs have been successful at describing a variety of material appearances and several works have developed interpretable parametriza-
tions [BS12], [KG13], [CPK06]. Those parametrizations are generally created to be artist-friendly with a representation focus on either a large set of materials or some detailed appearance effect. Unfortunately, analytical models are restricted in the range of appearances they can represent, thus triggering the idea of using measured BRDF. Indeed, with the increasing efficiency and accuracy of BRDF measurements as well as the decreasing cost of storage, datasets keep expanding and data-based reflection models have attracted a lot of attention [MPBM03], [FV14], [DJ18]. However, in contrast to analytical models, parametrization of measured BRDF models remains challenging. Pursuing this issue, three principal lines of investigation have emerged in recent years.

The first line of work propose to accommodate an analytical BRDF model to the measured ones [SJR18], [HP17], [BSH12]. Although quite easy to parameterize, they suffer from two drawbacks. Either the model is too simple to represent the richness of the measured data, or the model is complex and can encounter instability in the fitting procedure. The second line of work relies on user-based studies to design an interpretable parametrization space [WAKB09], [GXZ'13], [SGM'16]. They provide intuitive parametrization space but the test-subject study can be unwieldy to conduct. The last line of work, that includes the present investigation, makes use of machine learning methodologies. So far, while the offered solutions manage to fit the measured data very well, they do not produce interpretable parametrization solutions [MPBM03], [SSN18]. Close to our work and in an effort to ameliorate BRDF compression, two recent papers apply deep learning methods for faithful compression-decompression of the measured data [ZZW'21], [HGC'20]. Using previously established parametrizations ([SGM'16] and [GGPL18]), they confer some editing possibility to their work. However, those methods, for being effective compression tools, do not offer the capability of creating an interpretable parametrization space.

In this work, we leverage a Deep Neural Network (DNN) to learn a new interpretable parameter space for measured materials. We rely on a β-Variational AutoEncoder (β-VAE), a specific network architecture which enforces the creation of a disentangled latent space. The latent space is the space in which the compressed data lives. Disentangled means that each variable of the latent space is responsible for representing a specific generative factor and is insensitive to the factors represented by the other variables. Instead of using perceptual analysis through a cumbersome user study to define an interpretable control space, our set of variables is generated by the DNN through the training process. Then, in a second step, we explore the newly established parametrization space and we exploit our visual perception to interpret the generative factors learned by the network.

Expanding on an unsupervised DNN training, our method enjoys the precise fitting and appearance richness of machine learning approaches while offering the interpretability and control of analytical models without requiring crowd-sourced experiments.

The created interpretable parametrization space yields multiple editing applications. First, we can adjoin new parameters to the β-VAE learned set. The resulting refined set enlarges the range of possible appearance representations of our neural network. In addition, it enables the controllable post-tuning improvement of materials directly in image space to better match the initial measured BRDF. Indeed, our neural network output can be rendered and easily improved by changing a few values of our set of interpretable parameters. Because lighting and shape substantially influence our perception of appearance [SCW'21], being able to improve the quality of the reconstructed BRDF directly in the conditions it will be rendered comes as an important feature. Last, our interpretable parameter space allows more flexible interpolation between materials than random manifold exploring.

To sum up our contributions:

We propose a β-VAE based approach to establish a new interpretable editing parametrization space for measured BRDF models. Instead of resorting to a user study and collecting empirical observations, we train a DNN in an unsupervised way to efficiently generate a disentangled latent space of generative factors that are highly interpretable for humans.

We apply our newly found parametrization space for various tasks: reconstruction improvement, material editing with flexible interpolation as well as random manifold exploring and measured BRDF compression.

We provide two interfaces for material editing based on BRDF-Explorer [BS12], one, real-time, using our newly found parametrization space and the other for random manifold exploring. We can experience firsthand the higher control flexibility of our parametrization over the manifold exploration.

To the best of our knowledge, our work is the first report of a deep learning approach applied to create an interpretable parameter space for measured BRDF models.

The organization of the rest of the paper is as follows. First, we offer an overview of the state of the art in measured BRDF parametrization and other relevant current research. Then, we describe our method followed by our results before concluding.

2. Related Work

2.1. Measured BRDF Parametrization

Measured BRDF datasets. Because analytical modeling of material appearance is complex, measured BRDF models have always attracted a lot of attention. Measurement techniques have become more accurate and simpler [NJR15]. Nowadays, BRDF measurements can merely be accomplished using a smart phone camera and a flash [DAD'18]. Various datasets of BRDF measurements are available [FV14], [DJ18]. In this work, we decided to use the MERL dataset [MPBM03] as it offers a wide set of material measurement data. We will discuss in more details the specifics of the MERL BRDF data in Section 4.1.

Analytical model fitting. As touched upon earlier, finding a parametrization for measured BRDF models has been achieved using three main approaches. The first by fitting an analytical model to the measurements. The goal of those efforts are twofold, first to improve the accuracy of the analytical model by comparing them to measured data [HP17], [NDM05], [BP20], [BSH12], [BSN16]. Second is the possibility to edit measured BRDF by editing the fitted analytical model [SJR18], [DH15], [LBAD'06]. The primary challenge of this approach is to develop an analytical model
complex enough to represent the richness of the measured data, yet simple enough so that the fitting process remains stable.

User study investigations. A second method to parameterize measured BRDF calls for user studies and perceptual data gathering. This approach has the inherent advantage of offering a parametrization that is constitutively interpretable. It has been reported for a subset of appearance characteristics: by [WAKB09] and [PFG00] on glossy reflections or [GXZ*13] for translucency. It has also been studied for particular material types, like in [SWSR21] for metals. In a pioneering work, [SGM*16] offered an editing parametrization for all measured BRDF of the MERL dataset for all appearance characteristics. They relied on a Principal Component Analysis (PCA) dimension reduction of the initial data. Later, [SSN18] and [HGC*20] showed that dimension reduction of the measured BRDF with PCA requires retaining a large set of dimensions for good reconstruction of the material appearance, thus, making it impractical to use for parametrization. More recently, [SCW*21] offers an extensive user study to analyze the influence of shape and illumination on the perception of material appearance.

Machine learning. Machine learning approaches have been used to find manifold representations of measured BRDF data. In the original paper introducing the MERL database, a non linear dimension reduction techniques named charting [Bra02] is already being used [MPBM03]. By analyzing where groups of similar materials are located on the manifold, they can find paths between those groups to edit the material appearance. The approach to find the manifold as well as the interpolation between material has been refined in [SSN18] by relying on a Gaussian Process. However the editing is performed with pseudo-random manifold exploration. It is pseudo-random because the material clustering obtained on the manifold allows general control over the interpolation editing but does not allow precise control over specific appearance characteristics. In two recent works, deep learning approaches oriented toward compressing the input measurements have been presented [HGC*20], [ZZW*21]. Both offer some editing possibilities by using the parameterization offered in [GGPL18] and [SGM*16], but their approach does not allow the creation of an interpretable parametrization over the measured BRDF models. Current deep learning approaches lack interpretability.

In this work, we endeavor to alleviate the deep learning black box effect by finding an interpretable editing parametrization space for measured BRDF model. The newly generated editing parametrization is obtained without any user study, merely through the training of a DNN. Visual perception is applied in a second step to interpret the parametrization obtained by the network. Our goal is to find a parametrization space that offers at the same time, the ability to interpret the parametrization obtained by the network. Our goal is to find a parametrization space that offers at the same time, the ability to interpret the parametrization obtained by the network.

2.2. Generative Deep Neural Networks for materials

Deep Learning is a fast growing area of research. It has been very successfully applied to graphics in a wide variety of areas and specially rendering. [TFT*20] and [TTM*21] present a thorough review of the latest progress in neural rendering. In recent years, DNN solutions have been useful in generating Spatially-Varying-BRDF from pictures of a surface [DAD*18], [ZWX*20], [GLT*21]. To solve this problem, several works have focused their attention onto the latent space in order to improve the performances of their approach [HDMR21], [GLD*19]. However so far, none has discussed the possibility of finding a disentangled latent space. In our work, we also pay careful attention to the latent space, forcing its disentanglement in order to create an interpretable parametrization of the measured BRDF. Other examples of applications include [ZW21] where a deep learning approach is used to generate a fixed image with different material appearances, but the control over the generative process is done through a random manifold exploration and therefore lacks interpretability. In a very recent work, [DLGM22] improves on the latter work by offering interpretable editing of materials on images with a DNN. They rely on a crowdsourced experiment for perceptual data gathering.

2.3. Disentangled Deep Neural Networks

One major drawback of Deep Neural Networks is their behavioral opacity. To offer some interpretability on the learning process a growing amount of AI research tries to develop models offering a disentangled and compressed representation of the input data [HCT*21], [ZLLZ21], [HHLP20]. Indeed, disentangled representation of data usually have good human interpretability. Simply put, each variable encodes a generative factor that is perceptually meaningful. According to the definition of [RM18], a disentangled representation should embrace three elements, modularity, compactness and explicitness. Modularity depicts the fact that a generative factor is only encoded by a single latent variable and not shared by the others. Compactness signifies that no two variables represent the same generative factor. Explicitness represents the fact that all the generative factors are captured. Several different architectures have been described to find such a disentangled representation [HMP*17], [CDH*16], [KM18]. We picked the β-VAE architecture [HMP*17] for this work as it is the most studied one.

3. Background

In this section we offer some background on autoencoders, VAE and β-VAE architectures of DNN. [BHP*18], [HS06], [Wen18] and [KW19] thoroughly cover the latest developments on the subject.

3.1. Autoencoder

An autoencoder is a DNN architecture that is composed of two main blocks. First, an encoder \( g_\theta \) that takes an input \( x \) and compresses it into a latent vector of smaller dimension \( z \). The second block is called a decoder noted \( f_\theta \). It take the latent vector \( z \) as input and generates an output \( \hat{x} \). \( \phi \) and \( \theta \) are the parameters of the encoder and decoder, respectively.

\[
g_\theta(x) = z \quad \text{and} \quad f_\theta(z) = \hat{x}
\] (1)

The goal of training an autoencoder is to find \( \phi \) and \( \theta \) such that
the output $\hat{x}$ is as close as possible to $x$ along the norm $||.||$ defined by the user.

$$\hat{\phi}, \hat{\theta} = \text{argmin}_{\phi, \theta} (||x - f_\theta(g_\phi(x))||)$$  \hspace{1cm} (2)

The latent vector $z$ can then be used as a compressed representation of the input $x$. This architecture has been successfully applied to various tasks such as denoising or classification [VLBM08], [LCZW13] and has been used in [HGC’20] for BRDF compression. However, this architecture has limitations for generative applications. Indeed, because of the lack of constraints on the latent space, the continuity between the latent representations of the different inputs cannot be guaranteed, and the interpretability of each encoded dimension is not possible. To address the continuity issue, Kingma and Welling introduced the Variational Autoencoder (VAE) architecture [KW14].

3.2. Variational Autoencoder (VAE)

With a VAE, instead of encoding the input into a point in the latent space, it is mapped to a distribution. This distribution is named prior, noted $p_\alpha(z)$ and depends on some variables $\alpha$. For example, with a gaussian distribution, $\alpha$ is the mean and standard deviation. This prior distribution is chosen by the user. In our architecture, it is the encoder that generates the latent space. So the goal of the encoder, now a probabilistic encoder noted $g$ is to generate a distribution that matches the prior distribution. The latent vectors will then be sampled from the distribution generated by the encoder. We want to find the parameters $\phi$ such that for every input of the encoder $x$ the output $z$ follows the prior distribution: $g_\phi(z|x) \equiv p_\alpha(z)$. The decoder, now a probabilistic decoder noted $f_\theta$ takes the latent vector and tries to reconstruct the input as closely as possible. We want to find the parameters $\theta$ which maximize $f_\theta(x|z)$, the likelihood of reconstructing $x$ given $z$ which is sampled from the probabilistic encoder distribution.

In other words, in one hand we want the distribution given by the probabilistic encoder to be as close as possible to the prior distribution that is set by the user. On the other hand we want the decoder to maximize the likelihood distribution of reconstructing back any input $x$ from the corresponding latent sample $z$ where $z$ is obtained from the probabilistic encoder. The first element enforces a latent distribution, the second part enforces an identical reconstruction of the input. This can be formalized as follows [KW14]:

$$\hat{\phi}, \hat{\theta} = \text{argmin}_{\phi, \theta} (D_{KL}(g_\phi(z|x)||p_\alpha(z)) - \mathbb{E}_{z \sim q_\theta(z|x)} \log(f_\theta(x|z)))$$ \hspace{1cm} (3)

The first term enforces the probabilistic encoder to generate the prior distribution by finding $\phi$ which minimises the Kullback–Leibler Divergence, noted $D_{KL}$, between $p_\alpha(z)$ and $g_\phi(z|x)$. The second term enforces the reconstruction, that allows us to find $\theta$ that maximizes the log-likelihood of generating $x$ from $z$ where $z$ is sampled from $g_\phi(z|x)$.

3.3. $\beta$-Variational Autoencoder ($\beta$-VAE)

The VAE architecture and loss function ensure that the latent space is continuous by learning a latent distribution while trying to reconstruct the input as faithfully as possible. The $\beta$-VAE modifies the loss function defined in Equation 3 slightly through the inclusion of a multiplier term $\beta$ to the KL divergence part.

$$L_{\beta\text{-VAE}} = \beta * D_{KL}(q_\theta(z|x)||p_\alpha(z)) - \mathbb{E}_{z \sim q_\theta(z|x)} \log(p_\theta(x|z))$$ \hspace{1cm} (4)

$\beta = 1$ yields the original VAE loss function. Setting $\beta > 1$ puts more emphasis on the latent distribution for a better encoding. The most efficient encoding for conditionally independent factors is orthogonal. So emphasizing on the latent distribution encourages the generative factors to be represented in a disentangled fashion.

In the present work, we create a new DNN following a $\beta$-VAE architecture to learn a disentangled latent space over measured BRDF. In the next section, we will discuss the details of our architecture and training as well as the input pre-processing.

4. Method

4.1. Data Pre-processing

The MERL dataset [MPBM03] is composed of the measured BRDF of 100 isotropic materials. Measurements are densely sampled using Rusinkiewicz half-difference angles [Rus98]. The raw data is stored in a 3-dimensional table shaped (90, 90, 180) with high dynamic range RGB values for each entry. These high magnitude inputs can pose challenges for a DNN, so the following normalization scheme is adopted:

$$\rho = \frac{\log(\rho + S + \epsilon) - \log(\epsilon)}{\log(1 + \epsilon) - \log(\epsilon)}$$ \hspace{1cm} (5)

where $\rho$ is a single BRDF measurement, $\epsilon = 0.01$ is a small constant and $S$ is a scalar value used to render the MERL BRDF. $S$ is 10 for the red channel, 15 for the green channel, and 180 for the blue channel. Additionally, regions of the BRDF where the viewing direction or incident direction are below the horizon are set to 0 indicating an invalid entry [BS12]. The BRDF is treated as 180 RGB image slices shaped 90 $\times$ 90. To reduce the amount of data at the entry of our network, we discard 8/9 of the original slices at regular interval. This ratio is chosen following [HGC’20]. We now have 21 RGB images slices, producing 63 $\times$ 90 $\times$ 90 input size. The reconstruction of a full BRDF is obtained through linear interpolation between image slices.

4.2. Network Architecture

We expand on the autoencoder introduced by [HGC’20] to create a new $\beta$-VAE to learn a disentangled latent space. The proposed $\beta$-VAE architecture is illustrated in Figure 2. We choose a normal distribution $\mathcal{N}(0, 1)$ as a prior. The encoder $g_\phi$ generates a mean and a variance $\mu_x, \sigma_x$ for each input $x$. A single point $z$ is sampled from the normal distribution $\mathcal{N}(\mu_z, \sigma_z^2)$. 
Our proposed network is composed of several 2D convolutional layers. They all have a kernel size of $3 \times 3$, stride length of 2, and padding size of 1 except the last one of the decoder that has a kernel size of $4 \times 4$. Batch normalization and leaky ReLU activation are performed after each layer as indicated Figure 2. Nine residual blocks [HZRS16] with two 2D convolution layers (kernel size of $3 \times 3$, stride length of 2, and padding size of 1) with leaky ReLU activation are added. Then three fully connected layers are used to extract the mean and a standard deviation of the latent space distribution. The latent vector is chosen of size 8. To decode the latent vector, a point is sampled from the distribution and expanded by three fully connected layers. Before entering the first 2D convolutional layers of the decoder, the vector is reshaped to a $64 \times 12 \times 12$. Finally, after three residual blocks, and a last convolutional layer we get the output result in the same sliced BRDF format as the input.

\[
g_\theta(x) = [\mu, \sigma_i^2] \tag{6}
\]

\[
z \sim \mathcal{N} \left( \mu, \sigma_i^2 \right) \tag{7}
\]

\[
f_\theta(z) = \tilde{x} \tag{8}
\]

![Figure 2: Our $\beta$-VAE architecture.](image)

4.3. Implementation Details

Our $\beta$-VAE network is trained in an unsupervised way and implemented using PyTorch [PGM19]. The network is trained with the Adam optimizer [KB15], a learning rate of $3 \times 10^{-3}$, and batch size of 2 for 1000 epochs. Larger batch sizes failed in faithfully reproducing the color of the materials, leaving all of them brown. Equation 7 is replaced with the reparameterization trick [KW14] during training.

\[
z = \mu + \sigma \times \epsilon \quad \text{with} \quad \epsilon \sim \mathcal{N}(0, 1) \tag{9}
\]

The loss function is formulated as

\[
L^{(i)}_{\text{recon}} = ||\text{mask}(z^{(i)}) - \text{mask}(x^{(i)})|| \tag{10}
\]

\[
L^{(i)}_{KL} = -\frac{1}{2} \sum_{j=1}^{M} \left( 1 + \log \left( \sigma_j^{(i)} ight)^2 - \left( \mu_j^{(i)} ight)^2 - \left( \sigma_j^{(i)} \right)^2 \right) \tag{11}
\]

\[
\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \left( L^{(i)}_{\text{recon}} + \beta_{\text{norm}} L^{(i)}_{KL} \right) \tag{12}
\]

where $M$ is the dimensionality of the latent space, $N$ is the mini-batch size, $x$ is an input BRDF, $\tilde{x}$ is the reconstruction from Equation 8, $||\cdot||$ denotes the $L_2$ norm. The function mask returns the set of all valid BRDF entries. $\beta_{\text{norm}} = \frac{\beta}{N}$ where $\beta = 12$. $M = 8$ is the latent space size and $N = 63 \times 90 \times 90$ is the input size.

Concerning the dimension of the latent space, [HGC+20] have shown that $M = 10$ is a cut-off dimension: smaller latent space deteriorate the reconstruction greatly, while larger dimensions produce limited gains. We trained a DNN with $M = 8$ and $M = 16$. When $M = 16$, the explicitness of the network did not improve and we faced the same poor reconstruction with green materials. Furthermore, several parameters were not interpretable or did not seem to have noticeable control over the appearance and multiple parameters seemed to control the same appearance characteristic. Hence, we decided to use $M = 8$ to improve the disentanglement by further constraining the latent representation.

It can be noted here that by construction, the $\beta$-VAE architecture enables the compression/decompression of the input data. In our case, the $63 \times 90 \times 90$ input is compressed into an 8 elements vector. The storage of each material of the MERL dataset can be replace by the storage of each associated pair $(\mu, \sigma)$ and the decoder of our $\beta$-VAE. After training, the size of our model used to run our interfaces is 78.9 MB, the size of the latent vectors are negligible. For comparison, the size of one of the MERL measurement is 35 MB.

4.4. Interfaces

For practical handling of the disentangled parametrization, we developed an applicative interface based on BRDFExplorer [BS12] to which we added Pytorch C++ and CPython. As described earlier, for each BRDF measure, two vectors $\mu$ and $\sigma$ are produced by the encoder. Our new interface loads a pair of vectors $(\mu, \sigma)$ representing a BRDF and samples a vector $z$ from the normal distribution of parameters $(\mu, \sigma)$. We then propagate $z$ through the decoder of our $\beta$-VAE and render the output BRDF. Our interface runs in real-time and enables the user to modify each value of the latent vector $z$, leveraging the full strength of our parametrization. Indeed, each dimension of the latent vector $z$ represents one of our learned interpretable parameter over the measured BRDF. As described and illustrated in the Results Section, each dimension controls its own interpretable generative factor. Our interface includes sliders to mod-
Figure 3: Latent traversal of each variable of the latent space. We notice that each variable controls a distinct generative parameter. The names associated to each parameter are listed in Table 1. For metallic materials, unchanged values are set to the values obtained for the aluminium BRDF measurement. For diffuse materials, only Parameter n°5 is changed in the aluminium set for illustration purposes. For Parameter n°2, we chose the blue-fabric BRDF value for better contrast with Parameter n°3.

In addition, we put together a second interface that relies on manifold exploration for editing $z$. This interface is created for comparison sake with the previous related techniques [MPBM03], [SSN18], [ZZW*21]. The latent vector $z$ is mapped to a 2D embedding using Uniform Manifold Approximation and Projection (UMAP) [MHSG18]. UMAP is particularly useful because the algorithm preserves the global data structure and is invertible. A 2D point can be dragged over the manifold and mapped back to an 8-dimensions latent vector that is then passed through the decoder before rendering. We observe that this interface has limited editing control over generated material appearance as illustrated in another demonstration video available here.

5. Results

In this section, we illustrate the applicability of our approach and detail the outcomes. First and foremost, we show that we have successfully learned a highly interpretable editing parametrization space from the measured MERL BRDF dataset. Second, we illustrate that our interpretable disentangled parametrization affords the introduction of new generative parameters to the $\beta$-VAE learned set to expand the richness of possible material produced. Third, we demonstrate that our parametrization space allows controllable and flexible interpolation between materials as well as new material creation. We used Mitsuba renderer [Jak10] to render our images and we obtained some of the scenes used to illustrate our work from [Bit16].

5.1. Interpretable Editing Parametrization

Our goal is to learn an interpretable parametrization for measured BRDF model. To evaluate our parametrization, we perform a latent traversal of each of the 8 dimensions of our latent space. In other words, we modify the value of one of the latent parameter while leaving the others unchanged and render an image for each new value. In a second step, we exert our visual perception to express what we believe is the generative parameter being controlled by the latent variable. We exercise our perception only as a mean of interpretation of the parameters learned by our DNN. The learning process is unsupervised and does not require any human intervention.

The results of the latent traversal are presented Figure 3. We observe that each parameter controls its own generative factor. By relying on our visual perception and using the same terminology as in [BS12], we associate a name to each individual parameters. The naming association is detailed in Table 1.

The parameter naming process can be conducted easily and without ambiguity. Thus, we can confirm that the parametrization found is fully interpretable.

† Videos are fully anonymized. Link to the code for the interface has been removed to preserved anonymity
**Table 1:** Using the same terminology as in [BS12], we associate a name to the generative factor controlled by each parameter. The association is achieved through visual perception.

| Parameter no | Controlled generative factor                  |
|--------------|-----------------------------------------------|
| 1            | Diffuse color from blue to red                |
| 2            | Sheen                                         |
| 3            | Subsurface                                    |
| 4            | Clear coat                                    |
| 5            | Specular to Diffuse                           |
| 6            | Haziness                                      |
| 7            | Color lightness                               |
| 8            | Specular color from red to blue               |

5.2. New Parameters Creation for Expanded Material Appearance Richness

As detailed in the previous section, we have established an interpretable disentangled parametrization for the measured BRDF of the MERL dataset. The disentangled component is important as we can see that only two parameters control the color of the generated material: parameters no 1 and no 8, which represent respectively the diffuse and specular color from blue to red. In order to increase the color gamut of our original parametrization, we included two new parameters to the set learned by our DNN. These two new parameters control the diffuse and specular color from green to purple as observed in Figure 4.

The creation of these new parameters is achieved as follows. We generate two materials $M_1$ and $M_2$ from two sets of parameters $V_1$ and $V_2$ which have the same values for all parameters except for those controlling the color, i.e. parameters no 1 and no 8. As discussed previously, the two materials follow the RGB format. We replace the Green channel of $M_1$ with the Red channel of $M_2$, thus creating a new material $M$. Because parameters no 1 and no 8 of the set $V_2$ initially controlled the red diffuse and specular color of $M_2$, through this replacement, they now control the green diffuse and specular color of the new material $M$. Next, we add the parameters no 1 and no 8 of $V_2$ to the set $V_1$ creating a new set $V$ parametrizing material $M$. Those two parameters, now numbered 9 and 10 are responsible for controlling the green color for diffuse and specular reflections as illustrated in Figure 4.

This post-processing manual addition of parameters is only possible because the initial parametrization learned by our DNN is interpretable and disentangled. Indeed, the disentanglement feature forces the color to be controlled by two parameters only and the interpretability allows us to modify the original purpose of the parameters. The two newly created parameters allow us to broaden the range of generated material appearances. As illustrated Figure 5, we can now reproduce more faithfully the original material appearances of the MERL dataset by tuning our new set of parameters.

5.3. Flexible Material Editing

Previously available machine learning approaches relied on finding a path on a learned manifold for editing the measured materials. For comparison purposes we did project the latent space learned by our \(\beta\)-VAE onto a 2D manifold using the UMAP technique [MHSG18]. We then sampled $7 \times 7$ points on the found manifold and rendered the materials associated to each point. The results can be visualized in Figure 6. We identify some general appearance areas on the manifold. For example, the upper part is more diffuse and the bottom part is more specular. Navigating between the points allows some editing options. However, it proves quite challenging to interpolate
A. Benamira, S. Shah, S. Pattanaik / Interpretable Disentangled Parametrization of Measured BRDF with \( \beta \text{-VAE} \)

between material that are far apart on the manifold. Indeed, finding a path that interpolates faithfully the appearance between materials that are very different, i.e. far on the manifold, requires discovering a long path on the manifold. Moreover, it appears that on some paths small steps on the manifold can create abrupt changes on the material appearance as illustrated in the companion video demonstrating the UMAP interface (see previous link).

Figure 6: Previously available manifold exploration methods are impractical for editing. Top: UMAP manifold representation of the latent space. UMAP enables the representation of the latent space on a 2D plane. The latent vectors corresponding to the materials of the MERL dataset are represented by red dots, the \( 7 \times 7 \) sampled points are represented by black crosses. Bottom: Rendering of the sampled manifold points. We render the 49 sampled points with a directional light source. We can identify some general tendencies over the locations of similar materials. For example, brighter colors are on the left and darker colors on the right. Exploration of the manifold for material editing can be troublesome to control.

In contrast, our new parametrization provides much more flexibility and precise control over the interpolation of the materials as illustrated Figure 7 and Figure 8. Instead of finding a path on a manifold for which we cannot precisely know which generative factor will be modified, our parametrization allows the precise control over specific generative factors. Figure 8 and Figure 1 illustrate the precise editing and rich rendering capabilities of our approach.

Finally, our parametrization enables the creation of new materials not initially present in the dataset in a controlled fashion. Examples of new material can be seen Figure 9.

We believe our interpretable disentangled parametrization is a sensible solution toward closing the gap between analytical and measured BRDF models. It offers both good control over editing distinct generative appearance factors and high variance in the possible producible appearances.

6. Limitations and Future Work

We would like to discuss several limitations of our current parametrization that we would like to solve in future work. Also our set of parameters are interpretable, the practicability of the parametrization for artistic purposes is currently circumscribed. In future work, we would like to modify the training pipeline to make our parametrization more practical. We have identified, two issues with the current disentanglement of our parameter space that require further investigations, the modularity and explicitness of the parametrization. For the modularity, we identified leaks of some parameters through some others. For example we noticed that the color control is slightly leaking on to parameters \( n^7 \) and \( n^4 \). It can be observed in Figure 3. In future work we would like to prevent those leaks by improving our latent space representation like in [XGX*21] or [GH19]. The other limitation of our network is its explicitness. For example, our DNN cannot reproduce green by itself. The green reconstruction issue that our DNN suffers from initially was already present in [HGC*20]. They reported a RelAE of 0.086 for green-acrylic where we obtain 0.077 Figure 5. However, thanks to the interpretability of our parametrization, and this is a genuine novelty of our work, we can significantly improve the explicitness...
Figure 8: Material editing from hazy green specular (top left) to red diffuse (bottom right). First row we change the color from green to blue by modifying parameters $n^9$ and $n^{10}$ (decrease the green contribution) and parameters $n^1$ and $n^8$ (increase blue contribution). Second row we reduce the haziness by changing parameter $n^6$. Third row we transition from specular to diffuse using parameter $n^5$. Last row we modify the diffuse color from blue to red using parameters $n^1$ and $n^9$.

by manually adding dimensions to the latent space. We thus reach a RelAE of 0.028 with our final parametrization. Nonetheless, in future work, we would like to improve the natural explicitness of our network. Finally, this work is focused on the MERL dataset. Among many opportunities to consider for future developments, extending the approach to other material databases [DJ18], [FV14] would be an interesting option.

7. Conclusion

In an effort to combine the best of both worlds between the high variance of measured BRDF models and the high control parametrization of analytical BRDF models, we have developed a novel deep learning-based approach to create an interpretable disentangled parametrization space for measured BRDF. Our method relies on training a $\beta$-VAE and does not require a test-subject investigation. We have illustrated the visual interpretability of our parameters as well as the editing capabilities of our approach for reconstruction improvement or material editing. Furthermore, we have demonstrated that our set of learned parameters could be enriched with manual addition of parameters to extend the range of conceivable appearances. We believe that our approach leverages the great capacities of deep learning while removing one of its major drawbacks, the lack of interpretability.
Figure 9: Creation of new material using our parametrization. Our interpretable parametrization allows the creation of new materials not present in the original dataset.
