MaterialGAN: Reflectance Capture using a Generative SVBRDF Model

YU GUO, University of California, Irvine
CAMERON SMITH, Adobe Research
MILOŠ HASAN, Adobe Research
KALYAN SUNKAVALLI, Adobe Research
SHUANG ZHAO, University of California, Irvine

Fig. 1. We introduce a method to capture SVBRDF material maps from a small number of mobile flash photographs, achieving high quality results both on original and novel views. Our key innovation is optimization in the latent space of MaterialGAN, a generative model trained to produce plausible material maps; MaterialGAN thus serves as a powerful implicit prior for result realism. Here we show re-rendered views for several different materials under environment illumination. We use 7 inputs for these results (with 2 of them shown). (Please use Adobe Acrobat and click the renderings to see them animated.)

We address the problem of reconstructing spatially-varying BRDFs from a small set of image measurements. This is a fundamentally under-constrained problem, and previous work has relied on using various regularization priors or on capturing many images to produce plausible results. In this work, we present MaterialGAN, a deep generative convolutional network based on StyleGAN2, trained to synthesize realistic SVBRDF parameter maps. We show that MaterialGAN can be used as a powerful material prior in an inverse rendering framework: we optimize in its latent representation to generate material maps that match the appearance of the captured images when rendered. We demonstrate this framework on the task of reconstructing SVBRDFs from images captured under flash illumination using a hand-held mobile phone. Our method succeeds in producing plausible material maps that accurately reproduce the target images, and outperforms previous state-of-the-art material capture methods in evaluations on both synthetic and real data. Furthermore, our GAN-based latent space allows for high-level semantic material editing operations such as generating material variations and material morphing.

CCS Concepts: • Computing methodologies → Rendering.

Additional Key Words and Phrases: SVBRDF capture, generative adversarial network.

ACM Reference Format:

1 INTRODUCTION

Despite a few decades of effort in computer graphics and vision, capturing spatially-varying reflectance of real-world materials remains a challenging and actively researched task. Measurement methods have traditionally used custom hardware systems to densely sample illumination and viewing directions [Marschner et al. 1999; Matusik et al. 2003], followed by post-processing such as fitting parametric BRDF models [Ngan et al. 2005]. However, such approaches are restricted to laboratory conditions.

Recent work has explored methods for casual capture of spatially-varying BRDFs (SVBRDFs) using commodity hardware and in less constrained environments [Aittala et al. 2013, 2015; Francken et al. 2015].
These methods usually follow an inverse-rendering approach: they define a forward rendering model and optimize reflectance parameters so that the simulated appearance matches physical measurements under certain image metrics. With a small number of measured images, this approach is fundamentally under-constrained: there usually exist many material estimates capable of producing renderings that match the measurements, but many of these estimates can be unrealistic and may not generalize to novel illumination and viewing conditions. The solution to this problem has been to regularize the optimization using pre-determined material priors such as linear low-dimensional BRDF models \cite{Hui2017, Ren2011}. Moreover, as illustrated in Figure 10, MaterialGAN adapts to a wide range of SVBRDF samples ranging from diffuse to specular materials and near-stochastic textures with multiple distinct, complex materials.

Furthermore, our GAN-based latent space offers the ability to edit the latent vector in semantically meaningful ways (via operations like interpolation in the latent space) and generate realistic materials that go beyond the captured images. This is not possible with current material capture methods that do not afford any control over their per-pixel BRDF estimates.

2 RELATED WORK

Reflectance capture. Acquiring material data from physical measurements is the goal of a broad range of methods. Please refer to surveys \cite{Dong2019, Guarnera2016, Weyrich2009} for more comprehensive introduction to the related works.

Most reflectance capture approaches observe a material sample under varying viewing and lighting configurations. They differ in the number of light patterns required and their types such as moving linear light \cite{Gardner2003, Ren2011}. Gray code patterns \cite{Francken2009}, spherical harmonic illumination \cite{Ghosh2009}, and Fourier patterns \cite{Aittala2013}.

Methods have also been proposed for material capture “in the wild”, i.e., under uncontrolled environment conditions with commodity hardware, typically captured with a hand-held mobile phone with flash illumination. Some of these methods impose strong priors on the materials, such as linear combinations of basis BRDFs \cite{Hui2017, Xu2016} (where the basis BRDFs can come from the measured data \cite{Matusik2003}). Later work by Aittala et al. \cite{Aittala2015, Aittala2016} estimated per-pixel parameters of stationary spatially-varying SVBRDFs from two-shot and one-shot photographs. In the latter case, the approach used a neural Gram-matrix texture descriptor based on the texture synthesis and feature transfer work of Gatys \cite{Gatys2015, Gatys2016} to compare renderings with similar texture patterns but without pixel alignment.

More recently, deep learning-based approaches have demonstrated remarkable progress in the quality of SVBRDF estimates from single images (usually captured under flash illumination) \cite{Deschaintre2018, Li2017}. These methods train deep convolutional neural networks with large datasets of artistically created SVBRDFs, and with a combination of losses that evaluate the difference in material maps and renderings from the dataset ground truth.

Deschaintre \cite{Deschaintre2019} extended the single-shot approach to multiple images. The key idea is to extract features from the input images with a shared encoder, max-pooling the features and decoding the
final maps from the pooled features. This architecture has the benefit of being independent of the number of inputs, while also not requiring explicit light position information. In our experience, this approach produces smooth, plausible maps with low artifacts; however, re-rendering the maps tends to be not as close to the target measurements because the network cannot “check” its results at runtime. Moreover, we find that especially on real data, this method also has strong biases such as dark diffuse albedo maps and exaggerating surface normals (especially along strong image gradients that might be caused by albedo variations). We believe this is not due to any technical flaw; the method may be reaching the limit of what is possible using current feed-forward convolutional architectures and currently available datasets.

Gao et al. [2019] introduced an inverse rendering-based material capture approach that optimizes for material maps to minimize error with respect to the captured images. Since this is an under-constrained problem, they propose optimizing over the latent space of a learned material auto-encoder network to minimize rendering error. This approach has the benefit of explicitly matching the appearance of the captured image measurements, while also using the auto-encoder as a material “prior”. Moreover, the encoder and decoder are fully convolutional, which has the advantage of resolution independence. However, we find that the convolutional nature of this model also has the disadvantage of only providing local regularization and not capturing global patterns in the material, such as the long-range spatial patterns and correlations between the different material parameter maps. As a result, this method relies on previous methods (for example, Deschaintre et al. [2018]) to provide a good initialization, without which it can converge to poor results. In contrast, our MaterialGAN is a more globally robust latent space and produces higher quality reconstructions without requiring accurate initializations, though it is no longer resolution-independent.

**Generative adversarial networks.** GANs [Goodfellow et al. 2014b] have become extremely successful in the past few years in various domains, including images [Radford et al. 2015], video [Tulyakov et al. 2018], audio [Donahue et al. 2018], and 3D shapes [Li et al. 2019]. A GAN typically consists of two competing networks; a generator, whose goal is to produce results that are indistinguishable from the real data distribution, and a discriminator, whose job is to learn to identify generated results from real ones. For generating realistic images (especially of human faces), there has been a sequence of improved models and training strategies, including Progressive-GAN [Karras et al. 2018a], StyleGAN [2018b] and StyleGAN2 [2019]. StyleGAN2 in particular is the state-of-the-art GAN model and our work is based on its architecture, modified to output more channels.

Recently, GANs have also been used to solve inverse problems [Asim et al. 2019; Bora et al. 2017; O’Malley et al. 2019]. In computer graphics and vision, this work has focused on embedding images into the latent space, with the goal of editing the images in semantically meaningful ways via latent vector manipulations [Zhu et al. 2016]. This embedding requires solving an optimization problem to find the latent vector. More recent work such as Image2StyleGAN [Abdal et al. 2019b] and Image2StyleGAN++ [Abdal et al. 2019a] has looked at problem of embedding images specifically into the the StyleGAN latent space. While these methods focus on projecting portrait images into face-specific StyleGAN models, we find their analysis can be adapted to our problem. We build on this to propose a GAN embedding-based inverse rendering approach.

### 3 MATERIALGAN: A GENERATIVE SVBRDF MODEL

Generative Adversarial Networks [Goodfellow et al. 2014a] are trained to map an input from a latent space (often randomly sampled from a multi-variate normal distribution) to a plausible instance of the target distribution. In recent years, GANs have made remarkable progress in synthesizing high-resolution, photo-realistic images. Inspired by this progress, we propose MaterialGAN, a GAN that is trained to generate plausible materials, thus implicitly learning an SVBRDF manifold. MaterialGAN is based on the architecture of StyleGAN2 [Karras et al. 2019].

#### 3.1 Overview of StyleGAN and its latent spaces

StyleGAN2 [Karras et al. 2019] is an improvement of StyleGAN [Karras et al. 2018b] and is the state-of-the-art generative adversarial network (GAN) for image synthesis, especially for human faces. The architecture has several advantages over previous models like ProgressiveGAN [Karras et al. 2018a] and DCGAN [Radford et al. 2015]. For our purposes, the main advantage is that the model is not simply a black-box stack of convolution and upsampling layers, but has additional, more specific structure, allowing for much easier inversion (latent space optimization). The StyleGAN2 architecture starts with a learned constant $4 \times 4 \times 512$ tensor and progressively upsamples it to the final output target resolution via a sequence of convolutional and upsampling layers ($7$ in total to end with a final image resolution of $256 \times 256$). Given an input latent code vector $z \in \mathbb{Z} \subset \mathbb{R}^{512}$, StyleGAN2 transforms it through a non-linear mapping network of fully-connected layers into an intermediate latent vector $w \in W \subset \mathbb{R}^{512}$. The rationale for the introduction of the space of $W$ is that while $\mathbb{Z}$ requires (almost) every latent $z \in \mathbb{Z}$ to correspond to a realistic output, vectors $w \in W$ are free from this overly stringent constraint, which leads to a less “entangled” mapping, with more meaningful dimensions (see [Karras et al. 2018b, 2019] for more discussion). In the original StyleGAN, the vector $w \in W$ is mapped via a learned affine transformation to mean and variance “style” vectors that control adaptive instance normalizations (AdaIN) [Huang and Belongie 2017] that are applied before and after every convolution in the generation process (thus $7 \times 2 \approx 14$ times for a model of resolution $256 \times 256$). The statistics of the AdaIN normalizations caused the feature maps and output images of StyleGAN to suffer from droplet artifacts. StyleGAN2 removes the droplet artifacts entirely by replacing the AdaIN normalization layers with a demodulation operation which bakes the entire style block into a single layer while maintaining the same scale-specific control as StyleGAN. We construct a matrix $w^* \in W^* \subset \mathbb{R}^{512 \times 14}$ by replicating $w^*$ 14 times. During training and standard synthesis, the columns of $w^*$ are identical, and correspond to $w$. However, as we will discuss later (and similar to Abdal et al. [2019b]), we relax this constraint when optimizing for an embedding: $W^*$ thus becomes an extended latent space, more powerful than $W$ or $\mathbb{Z}$. Additionally, StyleGAN2 injects Gaussian noise, $\xi$, into each of the 14 layers of the generator. This noise gives StyleGAN2 the ability...
to synthesize stochastic details at multiple resolutions. Abdal et al. [2019a] make the observation that one can also treat these noise inputs \( \xi \) as a latent space \( \mathcal{N} \). Thus, combining these two spaces defines yet another latent space \( \mathcal{W}^* \mathcal{N} \).

3.2 MaterialGAN training

MaterialGAN was trained with the dataset provided by Deschaintre et al. [2018] (and also used in Gao et al. [2019]). They generated this dataset by sampling the parameters of procedural material graphs from Allegorithmic Substance Share to create an initial set of 155 high-quality SVBRDFs at resolution \( 4096 \times 4096 \). The dataset was augmented by blending multiple SVBRDFs and generating \( 256 \times 256 \) resolution crops at random positions, scales and rotations. The final dataset consists of around 200,000 SVBRDFs. For detailed information about the curation of dataset we refer the reader to Deschaintre et al. [2018]. Since pairs of SVBRDFs in the dataset were the same with only a slight variation, we selected 100,000 SVBRDFs. The maps for each SVBRDF are stacked in 9 channels (3 for albedo, 2 for normals, 1 for roughness, and 3 for specular albedo). We account for this by adapting the MaterialGAN architecture to output 9-channel normals, 1 for roughness, and 3 for specular albedo. We utilize a common BRDF model that involves a diffuse and a specular component using the microfacet BRDF with the GGX normal distribution [Walter et al. 2007]. Our unknown parameter vectors \( \theta = (a, n, r, s) \) encode the four per-pixel parameter maps: diffuse albedo \( a \), surface normal \( n \), roughness \( r \), and specular albedo \( s \). To recover the unknown parameter maps, we capture \( k \) images \( I_1, \ldots, I_k \).

The images can be reprojected into a common frontal view (which is straightforward with a known viewing configuration). We introduce a differentiable rendering operator \( R \) that takes as input the parameter maps as well as the viewing and lighting configurations, and synthesizes corresponding images of the material. Under this setup, our goal is to find values of the unknown parameters \( \theta \) so that renderings with these parameters match the measurements \( \{I_i\} \). In other words, we focus on solving the following optimization problem:

\[
\theta^* = \arg \min_{a, n, r} \sum_{i=1}^{k} L(R(\theta; I_i, C_i), I_i),
\]

where \( L \) is a loss function that measures the difference between the rendered image and the observed image.

4 SVBRDF CAPTURE USING MATERIALGAN

We utilize MaterialGAN, the powerful generative model described in the previous section, in a fundamentally new fashion: to capture SVBRDF maps. Specifically, we use MaterialGAN as a material prior for SVBRDF acquisition via an inverse rendering framework. Our goal is to estimate the SVBRDF parameter maps from one or a small number of photographs of a near-planar material sample. We utilize a common BRDF model that involves a diffuse and a specular component using the microfacet BRDF with the GGX normal distributions [Walter et al. 2007]. Our unknown parameter vectors \( \theta = (a, n, r, s) \) encode the four per-pixel parameter maps: diffuse albedo \( a \), surface normal \( n \), roughness \( r \), and specular albedo \( s \). To recover the unknown parameter maps, we capture \( k \) images \( I_1, \ldots, I_k \).

The images can be reprojected into a common frontal view (which is straightforward with a known viewing configuration). We introduce a differentiable rendering operator \( R \) that takes as input the parameter maps as well as the viewing and lighting configurations, and synthesizes corresponding images of the material. Under this setup, our goal is to find values of the unknown parameters \( \theta \) so that renderings with these parameters match the measurements \( \{I_i\} \). In other words, we focus on solving the following optimization problem:

\[
\theta^* = \arg \min_{a, n, r} \sum_{i=1}^{k} L(R(\theta; I_i, C_i), I_i),
\]

In theory, non-collocated lights, area lights or projection patterns (e.g. on an LCD or similar screen) can be used as well, and would require a straightforward modification to our forward rendering process.
where \( L \) is a loss function that measures the difference between the captured images, \( I_i \) and the renderings generated from the estimated SVBRDF parameters, \( R(\theta; I_i, C_i) \).

### 4.1 Incorporating the MaterialGAN prior

Eq. (1) is, in general, a challenging optimization to solve due to its under-constrained nature. Given a small number of input measurements, the optimization can overfit to the input, producing implausible maps that do not generalize to novel views and lighting. To overcome this challenge, we leverage the MaterialGAN prior: instead of directly optimizing for the parameter maps \( \theta \), we can optimize for a vector \( u \) in the MaterialGAN latent space and map (de)code this latent vector back into material maps \( \theta \). The optimization problem then becomes:

\[
\begin{align*}
    u^* &= \arg\min_u \sum_{i=1}^k L(R(\mathcal{G}(u); I_i, C_i), I_i),
\end{align*}
\]

where \( \mathcal{G} \) is the learned MaterialGAN generator. Given that both \( \mathcal{G} \) and \( R \) are differentiable operations, Eq. (2) can be optimized via gradient-based methods to estimate \( u^* \) and the corresponding SVBRDF maps \( \mathcal{G}(u^*) \). The above operation is similar to recent work on embedding images in the StyleGAN latent space [Abdal et al. 2019a,b]. The key difference is that we do not match material parameters directly, but evaluate their error through the rendering operator \( R(\cdot) \). To our knowledge, ours is the first approach to use a GAN latent space in combination with a rendering operator.

**Loss function.** We optimize Eq. 2 using a combination of a standard per-pixel L2 loss and a “perceptual loss” [Johnson et al. 2016] that has been shown to produce sharper results in image synthesis tasks:

\[
L(I, I') = \lambda_1 L_{\text{pixel}} + \lambda_2 L_{\text{percept}},
\]

where \( \lambda_1, \lambda_2 \) are hyperparameters. The perceptual loss is defined as:

\[
L_{\text{percept}}(I, I') = \sum_{j=1}^4 w_j^{\text{percept}} \| F_j(I) - F_j(I') \|_2^2,
\]

where \( F_1, \ldots, F_4 \) are the flattened feature maps corresponding to the outputs of VGG-19 layers \( \text{conv}1_1, \text{conv}1_2, \text{conv}3_2, \text{and} \text{conv}4_2 \) from a pre-trained VGG network [Simonyan and Zisserman 2015]. See section 4.3 for more details.

**Optimization details.** We convert the TensorFlow-trained MaterialGAN model to PyTorch, in which our optimization framework is implemented. We optimize Eq. 2 using the Adam optimizer in PyTorch, with a learning rate of 0.01. We set all other hyper-parameters to default values. Now that our basic optimization framework is set up, there remain two key ingredients to implement our GAN-based optimization framework (Eq. (2)): (i) the choice of latent space that we optimize \( u \) over, and (ii) our optimization strategy to minimize the objective function. In the following sections, we describe our approach, along with an empirical analysis of these design choices.

### 4.2 Latent space

As discussed in Sec. 3.1, StyleGAN2 (and consequently, MaterialGAN) has a number of potential latent spaces. In particular, MaterialGAN uses three different style latent spaces: the input latent code \( z \in \mathcal{Z} \), the intermediate latent code \( w \in \mathcal{W} \) and per-layer styles \( w^+ \in \mathcal{W}^+ \). StyleGAN2 also injects noise \( \xi \in \mathcal{N} \) into every layer of the network to generate stochastic variations. The typical forward generation process of the GAN only uses \( z \), with \( w \) being generated from \( z \) via a mapping network, and \( w^+ \) being generated from \( w \) via affine transformations. However, Abdal et al. [2019b] note that the space of \( \mathcal{Z} \) is too restrictive for accurate embedding of faces or other content into the GAN space. In other words, given the image of a human face, it is generally impossible to find a single \( z \in \mathcal{Z} \) such that the generated image closely matches the target. This remains the case even when extending the space to \( \mathcal{W} \), i.e., when searching for a \( w \) instead of a \( z \). The space \( \mathcal{W}^+ \), on the other hand, offers much stronger representative power. Our experiments on embedding material maps into MaterialGAN demonstrate that optimizing for \( \mathcal{W}^+ \) is also needed for MaterialGAN to accurately reproduce input maps. We demonstrate this in Figure 5, via an experiment where we embed a given material (with known material maps) into MaterialGAN. As shown in rows (2) and (3), maps generated by optimizing \( w^+ \in \mathcal{W}^+ \) contain more detail compared to those using \( w \in \mathcal{W} \).

On the other hand, some small-scale details are still missing. In fact, according to our experiments, only colors and large-scale features can be captured by the \( \mathcal{W}^+ \) space. For depicting high-frequency patterns, as demonstrated in rows (4) and (5) of Figure 5, we optimize for \( \xi \) in the pure-noise space \( \mathcal{N} \).
while leaving the small-scale ones to the noise vector \( \xi \).

Abdal et al. [2019a; 2019b] recommended using a two-stage setting by first optimizing \( \mathbf{w}^+ \) (with \( \xi \) fixed) and then \( \xi \) (with \( \mathbf{w}^+ \) fixed). In this approach, we need to go even further and optimize the noise vector \( \xi \) (instead of drawing it from multi-variate normal distributions). We note that optimizing for the noise component is even more important in MaterialGAN, compared to embedding faces in StyleGAN or StyleGAN2. We suspect that this is because with human faces, the distinction between large-scale features (e.g., eyes, noise, and mouth) and small-scale features (e.g., wrinkles) is very prominent, allowing the \( \mathcal{W}^+ \) space to focus mostly on the large-scale features while leaving the small-scale ones to the noise vector \( \xi \in \mathcal{N} \). In our case, the boundary between large-scale and small-scale material features is much less distinct. The physical scales of real-world materials vary in a continuous fashion, making it virtually impossible to assign them to only one of the \( \mathcal{W}^+ \) and \( \mathcal{N} \) spaces. We hypothesize that for this reason, we need to focus on both \( \mathcal{W}^+ \) and \( \mathcal{N} \) to achieve high-quality reconstruction of SVBRDF maps. Based on these empirical observations, estimating SVBRDF parameter maps from photographs using our pre-trained MaterialGAN boils down to solving the following optimization:

\[
\mathbf{u}^* = \arg \min_{\mathbf{w}^+ \in \mathcal{W}^+, \xi \in \mathcal{N}} \sum_{i=1}^{k} L(R(G(\mathbf{w}^+, \xi); I_i, C_i), I_i). \tag{5}
\]

Since there are two variables \( \mathbf{w}^+ \) and \( \xi \) that behave in a correlated fashion, a proper optimization strategy is crucial to achieve high-quality results. We now discuss our alternating two-step optimization method.

### 4.3 Optimization strategy

Abdal et al. [2019a; 2019b] recommended using a two-stage setting by first optimizing \( \mathbf{w}^+ \) (with \( \xi \) fixed) and then \( \xi \) (with \( \mathbf{w}^+ \) fixed). In our case, this approach does work in some cases but is not always the top-performing option. In addition to this strategy, we propose two alternatives, leading to three different optimization schemes:

1. **Strategy 1**: Optimize \( \mathbf{w}^+ \) first, then optimize \( \xi \);
2. **Strategy 2**: Jointly optimize both \( \mathbf{w}^+ \) and \( \xi \);
3. **Strategy 3**: Alternately optimize \( \mathbf{w}^+ \) and \( \xi \) for a small number (for example, 10) of iterations each.

Figure 6 shows a comparison of these strategies. All of them give reasonable results, but Strategy 1 is better suited for materials with strong large-scale features. Strategy 2 provides the fastest convergence because it allows the noise vector \( \xi \) to be modified from the very beginning. This, however, generally causes the optimization to use \( \xi \) for encoding higher-level features and is prone to overfitting. Finally, Strategy 3—a hybrid of Strategies 1 and 2—behaves in a more robust fashion than either of the previous strategies in most cases. In all the experiments, we use Strategy 3 for all the results in our paper. Additionally, our experiments indicate that it is desirable to use different VGG layer weights for the optimization of \( \mathbf{w}^+ \) and \( \xi \). The weights we are using are, for \( \mathbf{w}^+ \): [1/512, 1/512, 1/128, 1/64]; for \( \xi \): [1/64, 1/64, 1/256, 1/512].

**Noise optimization vs. post-refinement.** Instead of optimizing latent space \( \mathbf{w}^+ \) with noise \( \xi \), another option is to apply post-refinement (that is, pixel-space optimization without any latent space) after optimizing \( \mathbf{w}^+ \) only. However, the space \( \mathbf{w}^+ \) is too small to realistically match per-pixel detail: if optimizing \( \mathbf{w}^+ \) only, the resulting maps have significant artifacts. Adding post-refinement to such a result
We find that our method is robust to the initialization of the latent vector $\mathbf{w}$ with the lowest optimization error of Eq. (3) inverse-rendering optimizations and the remaining two (under novel illumination). Similar to previous work [Deschaintre et al. 2017], we use a paper frame to register the multiple images. We add markers to the frame to improve camera pose estimation. Using this process, we capture 28 physical samples with nine images per material, roughly covering the sample with $3 \times 3$ specular highlights. Unless otherwise specified, all our results use seven images for inverse-rendering optimizations and the remaining two (under novel lighting) for evaluating the results.

4.4 Initialization

We find that our method is robust to the initialization of the latent vectors. We experimented with using the same initial configuration—represented by the material produced by the mean $\mathbf{w}$ of our GAN training data (see Figure 8(a))—and found that is works well for most of the materials we tried (both synthetic and real). However, this initialization represents a material with a high roughness (reflecting a bias in our training data) and sometimes leads to errors when fitting highly specular / low roughness materials. Therefore, we add an additional low roughness initialization (see Figure 8(b)). In practice, given the captured images, we run our MaterialGAN optimization starting from both initializations and retain the result with the lowest optimization error of Eq. (3). All of our results in this paper followed this scheme.

5 RESULTS

Only a small subset of our results fits into the paper. Please see our supplemental material and video for more results.

Test data. For synthetic tests, we use several examples from the test set of Deschaintre [2018], as well as some from the Adobe Stock dataset [Li et al. 2018]. This gives a total of 39 synthetic results. For our real results, we use a hand-held mobile phone to capture images with flash, resulting in a collocated camera and point light illumination. Similar to previous work [Deschaintre et al. 2019; Hui et al. 2017], we use a paper frame to register the multiple images. We add markers to the frame to improve camera pose estimation. Using this process, we capture 28 physical samples with nine images per material, roughly covering the sample with $3 \times 3$ specular highlights. Unless otherwise specified, all our results use seven images for inverse-rendering optimizations and the remaining two (under novel lighting) for evaluating the results.

Inverse-rendering performance. Our optimization takes about 2 minutes to complete 2000 iterations on a Titan RTX GPU. In many cases, the results converge after 500 iterations, but we use 2000 everywhere for simplicity.

Testing on synthetic data. Figure 9 contains two synthetic results using our method, showing a close match both in maps and in novel view renderings. For more results, please refer to supplemental materials. We note that all methods perform better on synthetic data than on real data, possibly because of the exact BRDF model match and perfect calibration, and also because the synthetic test set, while distinct from the training set, is relatively similar in style.

5.1 Comparison with prior work on real data

Here we compare our method and Gao et al. [2019]. For more results and comparisons, including with Deschaintre et al. [2019], and including with and without initialization for ours and Gao’s method, please refer to supplemental materials. We show 10 real examples from our cell phone capture pipeline in Figure 10. Note that Gao’s method is significantly dependent on initialization, while the same is not true for our method. Therefore, in this figure, we show Gao’s result with initialization by Deschaintre et al. [2019], while our result is shown without initialization. Furthermore, note that we are initializing Gao’s method with the 2019 multi-input method by Deschaintre, which is a better initialization than the 2018 single-input method. Thus the baseline we are comparing against is, strictly speaking, even higher than what is published in Gao et al., and combines the two best methods published at this time. Generally, we find that our method produces cleaner maps and is less prone to overfitting (burn-in) than Gao’s, while producing more accurate re-renderings under original and novel lighting. Table 1 shows a quantitative evaluation of the re-rendering quality on novel lighting. As these novel views would be hard to match pixel-wise using any method, as they have never been observed, we use a perceptual method, specifically the Learned Perceptual Image Patch Similarity
**Fig. 10.** SVBRDF reconstruction on real data. We reconstruct SVBRDF maps from 7 inputs, and compare the resulting maps and images rendered under 2 novel views. Gao’s method [2019] initialized with Deschaintre’s [2019] direct predictions (denoted as “[Gao19]+”) tends to have complex reflectance burnt into the specular albedo map, leading to inaccurate predictions under novel views. Our method with simple initializations, in contrast, is less prone to such burn-ins and generally produces more accurate renderings under novel views. Please refer to Table 1 for more information on the quality of these renderings.
We find that the results from Deschaintre et al. [Deschaintre et al. 2019] have much less accurate re-rendering than either ours or Gao’s method, as they are not doing any optimization to precisely fit the target images. The mismatches we observe are definitely not due to simple scaling or gamma correction issues, as that would be consistent across examples; rather, we find that the method performs much better on synthetic examples that match the visual style of its training set. On the other hand, their method is fast and results tend to be clean and artifact-free, so they are very suitable for initialization of optimization methods.

5.2 Additional comparisons

Optimization with different initializations. In Figure 12, we compare our method to Gao’s with and without initialization by Deschaintre’s method in all 4 combinations, on a synthetic and a real example. This shows that Gao’s method more significantly depends on good initialization that ours (even though our method can still occasionally benefit).

Post-refinement. In general, the quality of our maps is sufficient after using our MaterialGAN-based optimization. However, Gao’s method introduced a post-refinement step, where the maps are further optimized without any latent space, and with at most minor regularization. Therefore, we also implement a similar post-refinement step. However, like good initialization, this post-refinement makes less of a difference in our method, and Gao’s method is more dependent on it, as it produces significantly blurry maps without it. This is shown in Figure 13; note the difference in sharpness of the maps.

Optimization with different numbers of input images. While most of our results are shown with 7 inputs, using two additional inputs for novel lighting evaluation, our method does work with various numbers of input images. We show 3 synthetic examples in Figure 14, with different numbers of inputs from 1 to 25. All the three examples are the same as used in Gao’s work. The errors of both reconstructed SVBRDF maps and novel-view renderings generally decrease with more input images, as is expected for an inverse-rendering method. In Figure 15, we compare real capture results with 1, 3, and 7 inputs, with and without initialization by Deschaintre’s method, and also include Gao’s results for 3 and 7 inputs (with
initialization). Our result remains plausible with 3 inputs, though artifacts do get reduced with more inputs. For all numbers of inputs, our result (with or without initialization) tends to be cleaner than Gao’s.

Editing operations. An additional advantage of the StyleGAN-based latent space is the ability to achieve semantically meaningful operations such as morphing, by interpolating two or more parent latent codes to create a hybrid offspring material. Morphing in latent space often preserves semantic features qualitatively better than naive interpolation in pixel space. Figure 16 and the supplemental video show morphing of a few real materials using linear interpolation in latent space, compared to the corresponding naive interpolation (linear in pixel space).

6 CONCLUSION AND FUTURE WORK

Discussion and limitations. While we believe our framework improves upon the state of the art, there are also some limitations. Our current BRDF model is shared by previous work, but certain common effects (layering on book covers, subsurface fiber scattering in woods, anisotropy in fabrics) are not correctly captured by it. An extension of our generative model and rendering operator would be possible, though the key challenge is finding high-quality training data for these effects.

Our assumption of almost flat samples will fail for materials with strong relief patterns, and will produce blurring or ghosting if there are obvious parallax effects in the aligned captured images. Strong self-shadowing or inter-reflections are also not currently handled. Solving for height instead of normal, with a more advanced rendering operator, may be able to resolve parallax effects and to correctly predict (and undo) shadowing effects from strong height variations.

Furthermore, more precise calibration may improve our accuracy. This would likely require knowledge of the cell phone hardware, and/or pre-calibration of its properties (e.g. flash light falloff, lens vignetting, and color processing properties).

Conclusion. We propose a novel method for acquiring SVBRDFs from a small number of input images, typically 3 to 7, captured using a hand-held mobile phone. We use an optimization framework that leverages a powerful material prior, based on a generative network, MaterialGAN, trained to synthesize plausible SVBRDFs. MaterialGAN learns correlations in SVBRDF parameters and provides local and global regularization to our optimization. This produces high-quality SVBRDFs that accurately reconstruct the input images, and because of our MaterialGAN prior, lie on a plausible material manifold. As a result, our reconstructions generalize better to novel views and lighting than previous state-of-the-art methods.
MaterialGAN: Reflectance Capture using a Generative SVBRDF Model

![SVBRDF maps](image1)

![Novel views](image2)

![SVBRDF maps](image3)

![Novel views](image4)

**Fig. 13. Per-pixel post-refinement.** Unlike Gao’s method, post-refinement via per-pixel optimization makes less of a difference in our method. Without post-refinement, [Gao19]+ (i.e., Gao’s method initialized with Deschaintre’s [2019] direct predictions) usually produces blurry results, as shown in the row marked as “[Gao19]+ (NR)”. Our method, on the contrary, does not rely nearly as heavily on post-refinement: Without it, our results are already quite sharp (see “Ours (NR)”), thanks to the generative power of our MaterialGAN. A zoomed-in version is attached below each SVBRDF map and novel-view image.

We believe that our work is only a first step toward GAN-based material analysis and synthesis and our experiments suggest many avenues for further exploration including improving material latent spaces and optimization techniques using novel architectures and losses, learning disentangled and editable latent spaces, and expanding beyond our current isotropic BRDF model.

**ACKNOWLEDGMENTS**

This research was started during Yu Guo’s internship at Adobe Research. We thank TJ Rhodes for help with material capture hardware setup. This work was supported in part by NSF IIS-1813553.

**REFERENCES**


Duan Gao, Xiao Li, Yue Dong, Pieter Peers, Kun Xu, and Xin Tong. 2019. Deep inverse rendering for high-resolution SVBRDF estimation from an arbitrary number of...


Fig. 15. **Performance using different numbers of input images (real data).** The quality of SVBRDF maps recovered by our method generally improves with more input images under both constant initialization (see “Ours”) and Deschaintre [2019] initialization (see “Ours+”).

Fig. 16. **Material interpolation.** Renderings of interpolations between two SVBRDFs recovered from real images using our method. Results on the left and right columns are obtained, respectively, using our GAN latent space and naïve linear interpolation.