Eurographics Symposium on Rendering 2020
C. Dachsbacher and M. Pharr
(Guest Editors)

Volume 39 (2020), *Number 4*

Photorealistic Material Editing Through Direct Image Manipulation

Károly Zsolnai-Fehér[1], Peter Wonka[2], Michael Wimmer[1]

[1]TU Wien
[2]KAUST

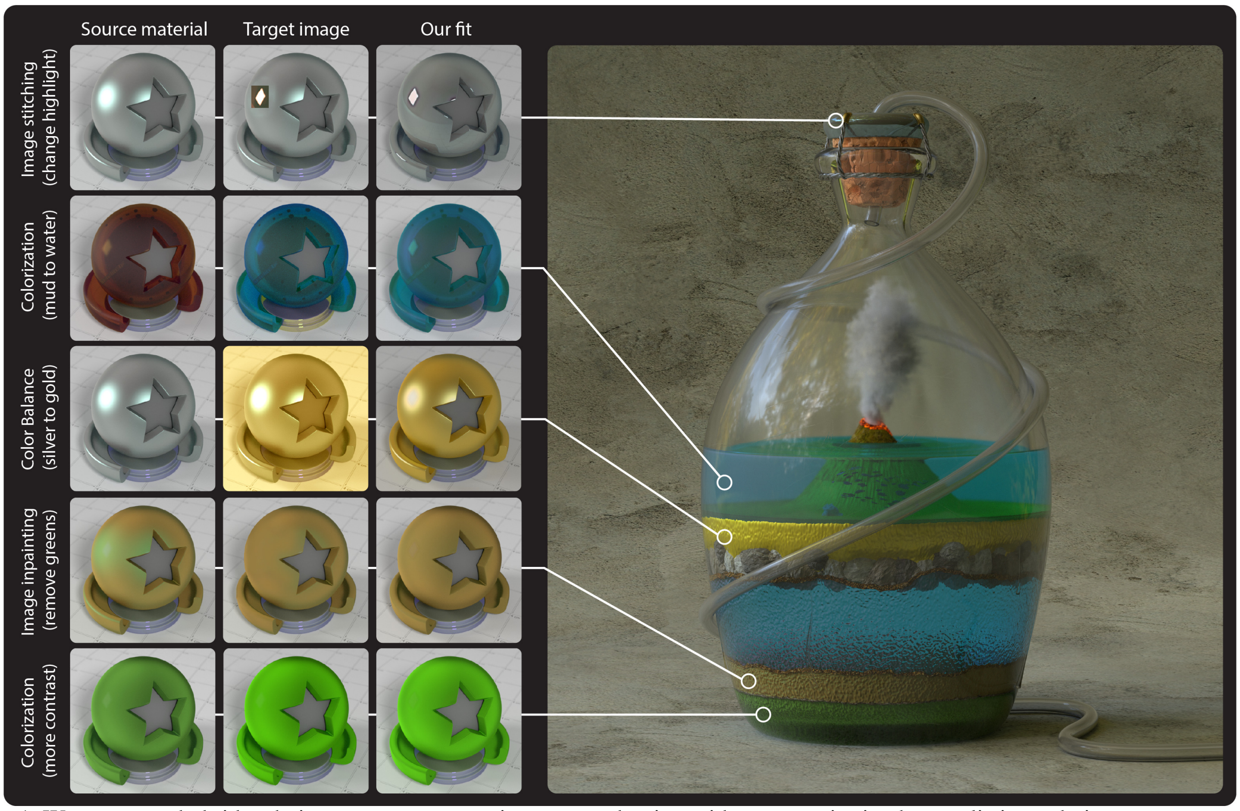
Source material
Target image
Our fit
Image stitching (change highlight)
Colorization (mud to water)
Color Balance (silver to gold)
Image inpainting (remove greens)
Colorization (more contrast)

Figure 1: We propose a hybrid technique to empower novice users and artists without expertise in photorealistic rendering to create sophisticated material models by applying standard image editing operations to a source image rendered in our reference scene (shown on the left). Then, in the next step, our method proceeds to find a photorealistic BSDF that, when rendered, resembles this target image. Our method generates each of the showcased fits within 20-30 seconds of computation time and is able to offer high-quality results even in the presence of poorly-executed edits (e.g., the background of the gold target image, the gold-colored pedestal for the water material and the stitched specular highlight above it). Scene: Reynante Martinez.

Abstract

Creating photorealistic materials for light transport algorithms requires carefully fine-tuning a set of material properties to achieve a desired artistic effect. This is typically a lengthy process that involves a trained artist with specialized knowledge. In this work, we present a technique that aims to empower novice and intermediate-level users to synthesize high-quality photorealistic materials by only requiring basic image processing knowledge. In the proposed workflow, the user starts with an input image and applies a few intuitive transforms (e.g., colorization, image inpainting) within a 2D image editor of their choice, and in the next step, our technique produces a photorealistic result that approximates this target image. Our method combines the advantages of a neural network-augmented optimizer and an encoder neural network to produce high-quality output results within 30 seconds. We also demonstrate that it is resilient against poorly-edited target images and propose a simple extension to predict image sequences with a strict time budget of 1-2 seconds per image.

CCS Concepts

• ***Computing methodologies*** $\rightarrow$ ***Neural networks; Rendering; Ray tracing;***

1. Introduction

The expressiveness of photorealistic rendering systems has seen great strides as more sophisticated material models became available for artists to harness. Most modern rendering systems offer a node-based shader tool where the user can connect different kinds of material models and perform arbitrary mathematical operations over them (e.g., addition and mixing), opening up the possibility of building a richer node graph that combines many of the more rudimentary materials to achieve a remarkably expressive model. These are often referred to as "principled" shaders and are commonly used within the motion picture industry [BS12]. However, this expressiveness comes with the burden of complexity, i.e., the user has to understand each of the many parameters of the shader not only in isolation, but also how they influence each other, which typically requires years of expertise in photorealistic material modeling. In this work, we intend to provide a tool that can be used by a wider target audience, i.e., artists and novices that do not have any experience creating material models, but are adept at general-purpose image processing and editing. This is highly desirable as human thinking is inherently visual and is not based on physically-based material parameters [RSB*02, Whi89]. We propose a workflow in which the artist starts out with an image of our material test scene and applies classic image processing operations to it. Our key observation is that even though this processed target image is often not physically achievable, in many cases, a photorealistic material model can be found that is remarkably close to it (Fig. 2). These material models can then be easily inserted into already existing scenes by the user (Fig. 3).

In summary, we present the following contributions:

- An optimizer that can rapidly match the target image when given an approximate initial guess.
- A neural network to solve the adjoint rendering problem, i.e., take the target image as an input and infer a shader that produces a material model to approximate it.
- A hybrid method that combines the advantages of these two concepts and achieves high-quality results for a variety of cases within 30 seconds.
- A simple extension of our method to enable predicting sequences of images within 1-2 seconds per image.

We provide our pre-trained neural networks and the source code for the entirety of this project.

2. Previous Work

2.1. Material Acquisition

A common workflow for photorealistic material acquisition requires placing the subject material within a studio setup and using measurement devices to obtain its reflectance properties. To import this measured data into a production renderer, it can be either used as-is, can be compressed down into a lower-dimensional representation [PRJ*13, RJGW19, WAA*00] or approximated through an analytic bidirectional scattering distribution function (BSDF) model [PdMJ14]. Due to the large body of research works in this area, we relate our method to a few commonly used works and refer the interested reader to the appropriate survey papers for more information [WdBKK15, GGG*16]. Many recent endeavors improve the cost efficiency and convenience of this acquisition step by only requiring photographs of the target material [AWL*15, AAL16, DAD*18, LDPT17, LSC18, GRR*17] while still requiring physical access to these source material samples, while precomputed BSDF databases offer an enticing alternative where the user can choose from a selection of materials [Mat03, DJ18]. We aim to provide a novel way to exert direct artistic control over these material models. Our method can be related to inverse rendering [MG98, RH01] and appearance computation [WDR13] approaches, where important physical material properties are inferred from a real photograph with unknown lighting conditions. In our work, the material test scene contains a known lighting and geometry setup, but in return, enables not only the rapid discovery of new materials, but artistic control through standard and well-known image-space editing operations. Our method can also be thought of as a specialized version of recent differentiable rendering approaches [LHJ19, CLZ*18] that is designed for capturing and reproducing intricate details in material appearance.

2.2. Material Editing

To be able to efficiently use the most common photorealistic rendering systems, an artist is typically required to have an understanding of physical quantities pertaining to the most commonly modeled phenomena in light transport, e.g., indices of refraction, scattering and absorption albedos and more [STPP09, BS12, NSR17]. This modeling time can be cut down by techniques that enable editing bidirectional reflectance distribution function (BRDF) models directly within the scene [BAOR06, CPWAP08, SZC*07], however, with many of these methods, the artist is still required to understand the physical properties of light transport, often incurring a significant amount of trial and error. Instead of editing the materials directly [SJR18], other techniques enable editing secondary effects, such as caustics and indirect illumination within the output image [SNM*13, BAEDR08]. Other efficient material editing workflows also open up the possibility of material remapping [SKWW17], retargeting [ATDP11], editing measured SVBRDFs [PL07] and rapid relighting previously rendered scenes [WCPL*08, NRH04, WTL04]. Reducing the expertise required for material editing workflows has been a subject to a large volume of research works: an intuitive editor was proposed by pre-computing many solutions to enable rapid exploration [HR13], carefully crafted material spaces were derived to aid the artist [SGM*16, SSN18, LMS*19], and learning algorithms have been proposed to create a latent space that adapts to the preferences of the user [ZFWW18]. Other image-based editing methods open up the possibility of editing BSSRDFs [RCP14] or SVBRDFs through an inferred albedo map [DTPG11]. We also endeavored to create a solution that produces the desired results *rapidly* by looking at a non-physical mockup image, requiring expertise only in 2D image editing, which is considered to be common knowledge by nearly all artists in the field. Generally, BRDF relighting methods are preferable when in-scene editing is a requirement [LCY*17, NSRS13], otherwise, we recommend using our proposed technique in the case of one sought material to moderate-scale problems and Gaussian Material Synthesis (GMS) [ZFWW18] for mass-scale material synthesis.

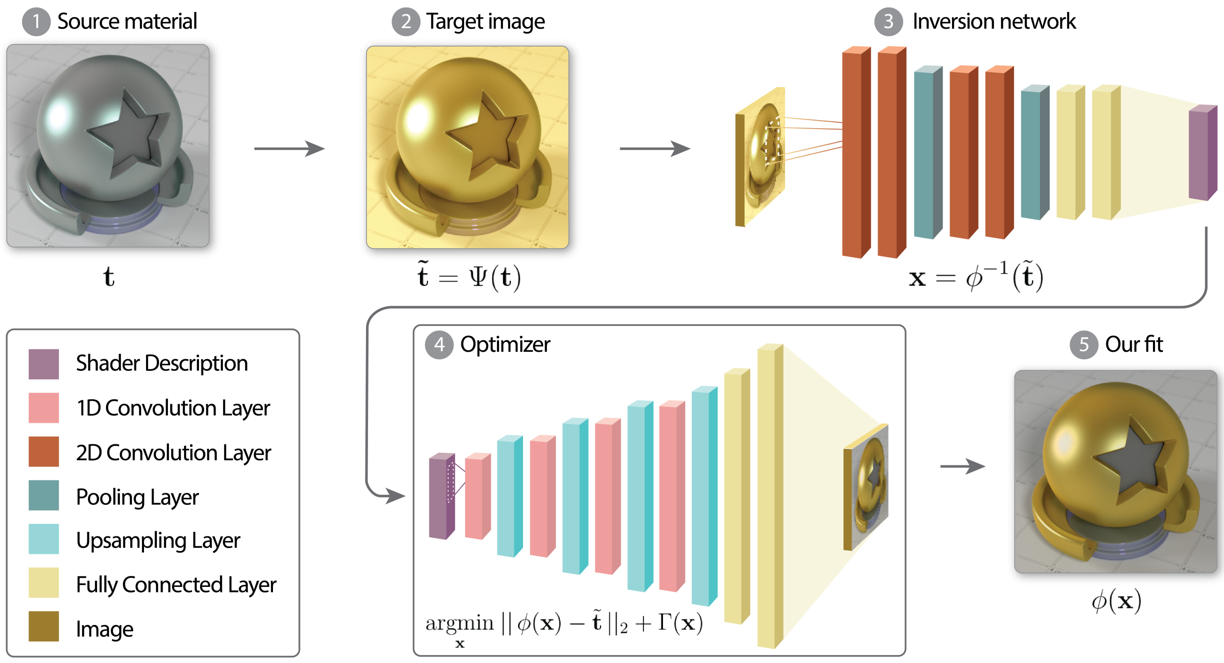

Figure 2: Our proposed hybrid technique offers an intuitive workflow where the artist takes a source material (❶) and produces the target image by applying the desired edits to it within a 2D raster image editor of their choice (❷). Then, one or more encoder neural networks are used to propose a set of approximate initial guesses (❸) to be used with our neural network-augmented optimizer (❹), which rapidly finds a photorealistic shader setup that closely matches the target image (❺). The artist then finishes the process by assigning this material to a target object and renders the final scene offline.

2.3. Neural Networks and Optimization

Optimization is present at the very core of every modern neural network: to be able to minimize the prescribed loss function efficiently, the weights of the networks are fine-tuned through gradient descent variants [Bot10, RM51] or advanced methods that include the use of lower-order moments [KB14], while additional measures are often taken to speed up convergence and avoid poor local minima [SMDH13, Goh17]. Similar optimization techniques are also used to generate the model description and architecture of these neural networks [ZL16, EMH18], or the problem statement itself can also be turned around by using learning-based methods to discover novel optimization methods [BZVL17]. In this work, we propose two combinations of a neural network and an optimizer – first, the two can be combined *indirectly* by endowing the optimizer with a reasonable initial guess, and *directly* by using the optimizer that invokes a neural renderer at every function evaluation step to speed up the convergence by several orders of magnitude (steps ❸ and ❹ in Fig. 2). This results in an efficient two-stage system that is able to rapidly match a non-physical target image and does not require the user to stay within a prescribed manifold of artistic editing operations. In a related approach, Zhu et al. [ZKSE16] have also used the first mentioned combination. Their method uses a generative model to synthesize images, restricting the space of possible image editing operations, whereas our technique seeks a parameter setup to be used with a principled shader, allowing more general image edits, but also requiring a more elaborate scheme to provide robust results. The use of an optimizer to refine a neural network prediction can also be applied to reflectance capture [KCW*18] and image-based material estimation [GLD*19], while a multi-network variant can be used to perform texture synthesis [HDR19]. In summary, our technique handles marked up image inputs that are outside of the training domain, and uses an optimizer to refine a collection of neural network predictions to yield a low-dimensional material representation. It also supports interactive workflows where rapid iteration is required and is independent of the underlying BSDF representation as long as the associated neural renderer works reliably.

3. Overview

Many trained artists are adept at creating new photorealistic materials by engaging in direct interaction with a principled shader. This workflow includes adjusting the parameters of this shader and waiting for a new image to be rendered that showcases the appropriate output material. If at most a handful of materials are sought, this is a reasonably efficient workflow, however, it also incurs a significant amount of rendering time and expertise in material modeling. Our goal is to empower novice and intermediate-level users to be able to reuse their knowledge from image processing and graphic design to create their envisioned photorealistic materials (where the degree of photorealism is determined by the capabilities of the shader).

Instead of using a photograph of a material sample as an input, in this work, we set up a *material test scene* that contains a known lighting and geometry setup, and a fixed *principled shader*

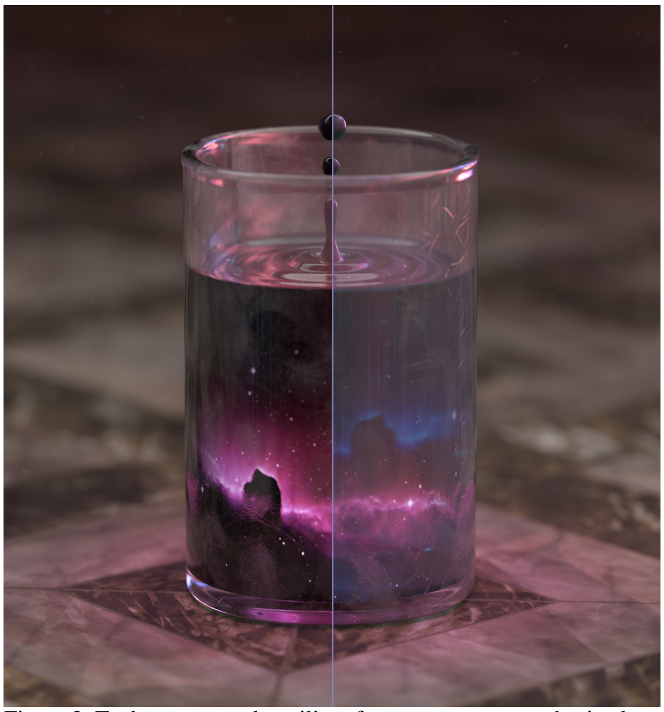
Figure 3: To demonstrate the utility of our system, we synthesized a new material using the material test scene shown in Fig. 2 and then deployed it into an already existing scene using Blender and Cycles. In this scene, we made a material mixture to achieve a richer and foggier nebula effect inside the glass. Left: theirs, right: 50% theirs, 50% ours. Scene: Reynante Martinez.

with a vector input of $x \in \mathbb{R}^m$. We chose the scene to be one that artists working in the industry are already familiar with to make sure that results on this scene can be intuitively transferred to the desired production scene. We use the shader from Zsolnai-Fehér et al. with $m = 19$ [ZFWW18], which contains many albedo-related parameters and is able to represent the most commonly used diffuse, glossy, specular and translucent materials with varying roughness and volumetric absorption coefficients. Each parameter setup of this shader produces a different material model when rendered. In our workflow, the user is offered a variety of images, and chooses one desired material model as a starting point. Then, the user is free to apply a variety of image processing operations on it, e.g., colorization, image inpainting, blurring a subset of the image and more. Since these image processing steps are not grounded in a physically-based framework, the resulting image is not achievable by adjusting the parameters in the vast majority of cases. However, we show that our proposed method is often able a produce a photorealistic material that closely matches this target image.

Solution by optimization. When given an input image $\mathbf{t} \in \mathbb{R}^p$, it undergoes a series of transformations (e.g., colorization, image inpainting) as the artist produces the target image $\tilde{\mathbf{t}} = \Psi(\mathbf{t})$, where $\Psi : \mathbb{R}^p \to \mathbb{R}^p$. Then, an image is created from an initial shader configuration, i.e., $\phi : \mathbb{R}^m \to \mathbb{R}^p$, where m refers to the number of parameters within the shader and p is the number of variables that describe the output image (in our case $p = 3 \cdot 410^2$ is used with the range of 0-255 for each individual pixel). This operation is typically implemented by a global illumination renderer. Our goal is to find an appropriate parameter setup of the principled shader $\mathbf{x} \in \mathbb{R}^m$ that, when rendered, reproduces $\tilde{\mathbf{t}}$ (note that in order to conform to artist expectations, both $\mathbf{t}$ and $\tilde{\mathbf{t}}$ are assumed to be in image space, i.e., tone-mapped). Generally, this is not possible as a typical Ψ leads to images that cannot be perfectly matched through photorealistic rendering. However, surprisingly, we can often find a configuration $\mathbf{x}$ that produces an image that closely resembles $\tilde{\mathbf{t}}$ through solving the minimization problem

$$\begin{aligned} &\underset{\mathbf{x}}{\text{argmin}} && \|\phi(\mathbf{x}) - \tilde{\mathbf{t}}\|_2, \\ &\text{subject to} && \mathbf{x}_{\text{min}} \leq \mathbf{x} \leq \mathbf{x}_{\text{max}}, \end{aligned} \tag{1}$$

where the constraints stipulate that each shader parameter has to reside within the appropriate boundaries (i.e., $0 \leq x_i \leq 1$ for albedos or $x_j \geq 1$ for indices of refraction where $x_i, x_j \in \mathbf{x}$). To be able to benchmark a large selection of optimizers, we introduce an equivalent alternative formulation of this problem where the constraints are reintroduced as a barrier function $\Gamma(\cdot)$, i.e.,

$$\underset{\mathbf{x}}{\text{argmin}} \left(\|\phi(\mathbf{x}) - \tilde{\mathbf{t}}\|_2 + \Gamma(\mathbf{x}) \right), \text{ where}$$
$$\Gamma(\mathbf{x}) = \begin{cases} 0, & \text{if } \mathbf{x}_{\text{min}} \leq \mathbf{x} \leq \mathbf{x}_{\text{max}}, \\ +\infty, & \text{otherwise.} \end{cases} \tag{2}$$

In a practical implementation, the infinity can be substituted by a sufficiently large integer. This formulation enabled us to compare several optimizers (Table 3 in Appendix B), where we found Nelder and Mead's simplex-based self-adapting optimizer [NM65] to be the overall best choice due to its ability to avoid many poor local minima through its contraction operator and used that for each of the reported results throughout this manuscript.

Nonetheless, solving this optimization step still takes several hours as each function evaluation invokes ϕ, i.e., a rendering step to produce an image, which clearly takes too long for day-to-day use in the industry. We introduce two solutions to remedy this limitation, followed by a hybrid method that combines their advantages.

Neural renderer. To speed up the function evaluation process, we replace the global illumination engine that implements ϕ with a neural renderer [ZFWW18]. This way, instead of running a photorealistic rendering program at each step, our optimizer invokes the neural network to predict this image, thus reducing the execution time of the process by several orders of magnitude, in our case, from an average of 50 seconds to 4ms per image at the cost of restricting the material editing to a prescribed scene and lighting setup. Because of the lack of a useful initial guess, this solution still requires many function evaluations and is unable to reliably provide satisfactory solutions.

Solution by inversion. One of our key observations is that an approximate solution can also be produced *without* an optimization step by finding an appropriate inverse to ϕ: since ϕ is

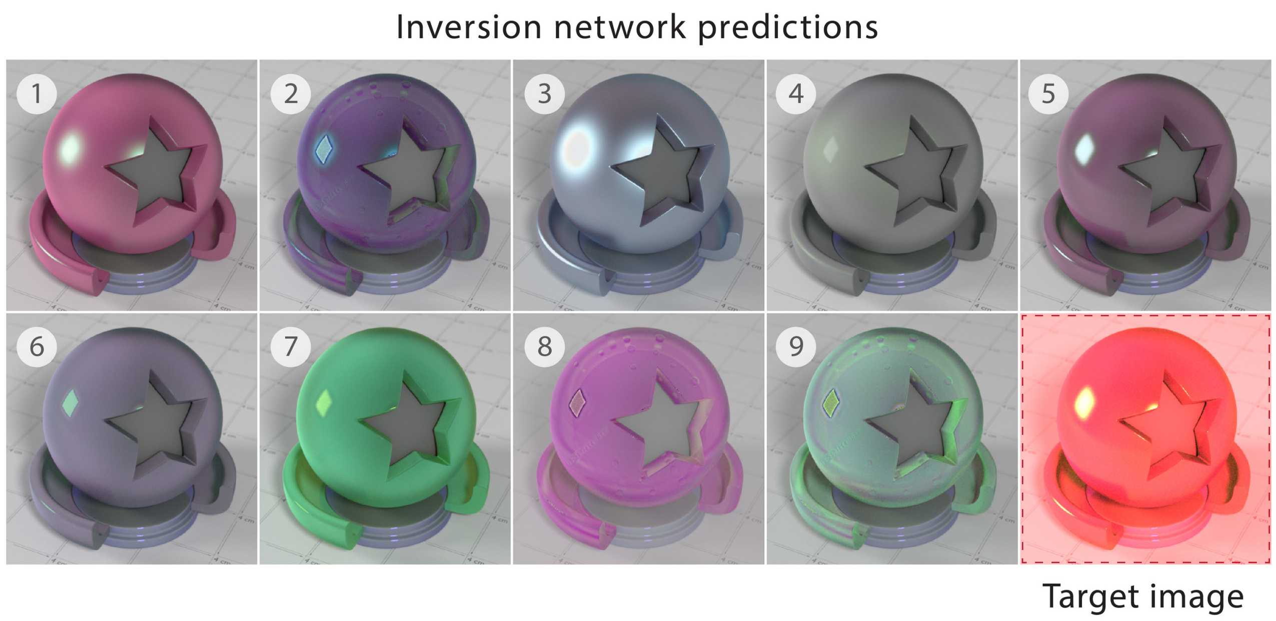

Figure 4: Whenever the target image (lower right) strays too far away from the images contained within their training set, our 9 inversion networks typically fail to provide an adequate solution and potentially predict results outside the feasible region (❷, ❽, ❾). However, using our "best of n" scheme and our hybrid method, the best performing prediction of our neural networks can be used to equip our optimizer with an initial guess, substantially improving its results.

realized through a decoder neural network (i.e., neural renderer) that produces an image from a shader configuration, ϕ^{-1}, its inverse, can be implemented as an *encoder* network that takes an image as an input and predicts the appropriate shader parameter setup that generates this image. This adjoint problem has several advantages: first, such a neural network can be trained on the same dataset as ϕ by only swapping the inputs and outputs and retains the advantageous properties of this dataset, e.g., arbitrarily many new training samples can be generated via rendering, thereby loosening the ever-present requirement of preventing overfitting via regularization [SHK*14, NH92, ZH05]. Second, we can use it to find a solution *directly* through $\mathbf{x} \approx \phi^{-1}(\tilde{\mathbf{t}})$ without performing the optimization step described in (1-2). As the output image is not produced through a lengthy optimization step, but is inferred by this encoder network, this computes in a few milliseconds. We will refer to this solution as the *inversion network* and note that our implementation of ϕ^{-1} only approximately admits the mathematical properties of a true inverse function. We also discuss the nature of the differences in more detail in Section 4. We have trained 9 different inversion network architectures and found that typically, each of them performs well on a disjoint set of inputs. Our other key observation is that because we have an atypical problem where the ground truth image ($\tilde{\mathbf{t}}$) is available and each of the candidate images can be inferred inexpensively (typically within 5 milliseconds), it is possible to compute a "best of n" solution by comparing all of these predictions to the ground truth, i.e.,

$$\mathbf{x} = \phi^{-1}_{(i)}(\tilde{\mathbf{t}}), \text{ where } i = \underset{j}{\operatorname{argmin}} \, ||\phi(\phi^{-1}_{(j)}(\tilde{\mathbf{t}})) - \tilde{\mathbf{t}}||_2, \quad (3)$$

where $\phi^{-1}_{(i)}$ denotes the prediction of the i-th inversion network, $j = (1, \ldots, n)$, and in our case, $n = 9$ was used. This step introduces a negligible execution time increase and in return, drastically improves the quality of this inversion process for a variety of test cases. However, these solutions are only approximate in cases where the target image strays too far away from the training data (Fig. 4). In Appendix A we report the structure of the neural networks used in this figure.

Hybrid solution. Both of our previous solutions suffer from drawbacks: the optimization approach provides results that resemble $\tilde{\mathbf{t}}$ but is impracticable due to the fact that it requires too many function evaluations and gets stuck in local minima, whereas the inversion networks rapidly produce a solution, but offer no guarantees when the target image significantly differs from the ones shown in the training set. We propose a hybrid solution based on the knowledge that even though the inverse approach does not provide a perfect solution, since it can produce results instantaneously that are significantly closer to the optimum than a random input, it can be used to endow the optimizer with a reasonable initial guess. This method is introduced as a variant of (2) where $\mathbf{x}_{\text{init}} = \phi^{-1}(\tilde{\mathbf{t}})$ and a more detailed description of this hybrid solution is given below in Algorithm 1. Additionally, this technique is able to not only provide a "headstart" over the standard optimization approach but was also able to find higher quality solutions in all of our test cases.

Predicting image sequences. A typical image editing workflow

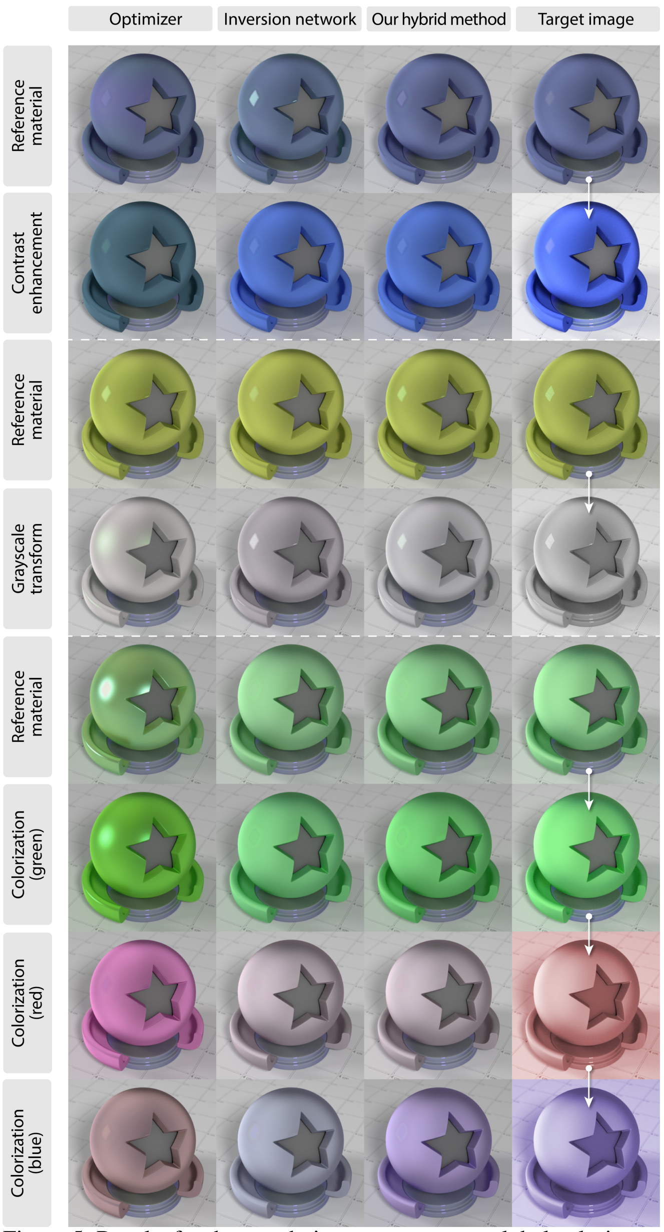

Figure 5: Results for three techniques on common global colorization operations including saturation increase and grayscale transform. The "reference material" labels showcase materials that can be obtained using our shader and would be picked by a user from a random gallery as starting point for the editing operation. The arrows indicate which images were the input and output of image-processing operations performed by the user. The results of the three methods in reference-material rows indicate how well the methods can reproduce materials that are actually exactly reproducible by the shader.

Algorithm 1 Photorealistic Material Editing

```
1: **Given** $\mathbf{t}$, $\phi(\cdot)$, $[\phi^{-1}_{(1)}(\cdot),\ldots,\phi^{-1}_{(n)}(\cdot)]$, $\mathbf{x}_{\min}$, $\mathbf{x}_{\max}$
2: $\tilde{\mathbf{t}} \leftarrow \Psi(\mathbf{t})$                          ▷ Obtain target image
3: **for** $i \leftarrow 1$ to $n$ **do**             ▷ Predict with $n$ inversion networks
4:     Compute each $\phi^{-1}_{(i)}(\tilde{\mathbf{t}})$
5: **Find** i = $\text{argmin}_{j\in 1..n}\, ||\phi(\phi^{-1}_{(j)}(\tilde{\mathbf{t}})) - \tilde{\mathbf{t}}||_2$  ▷ Find best candidate
6: Define $\mathbf{x}_{\text{init}} \leftarrow \phi^{-1}_{(i)}(\tilde{\mathbf{t}})$
7: Define $f_1(\mathbf{x}) = \mathbf{x}_{\max} - \mathbf{x}$                  ▷ Set up constraints
8: Define $f_2(\mathbf{x}) = \mathbf{x} - \mathbf{x}_{\min}$
9: Define $\mathcal{C} = \{\mathbf{x} \mid f_i(\mathbf{x}) \geq \mathbf{0},\ i = 1,2\}$     ▷ Construct feasible region
10: Define $\Gamma(\mathbf{x}) = \begin{cases} 0, & \text{if } \mathbf{x} \in \mathcal{C}, \\ +\infty, & \text{otherwise} \end{cases}$   ▷ Construct barrier
11: **Initialize** optimizer with $\mathbf{x}_{\text{init}}$
12: **Minimize** $\text{argmin}_{\mathbf{x}} \left(||\phi(\mathbf{x}) - \tilde{\mathbf{t}}||_2 + \Gamma(\mathbf{x})\right)$ ▷ Refine initial guess
13: Display $\phi(\mathbf{x})$ to user
```

takes place within a raster graphics editor program where the artist endeavors to find an optimal set of parameters, e.g., the kernel width σ in the case of a Gaussian blur operation to obtain their envisioned artistic effect. This process includes a non-trivial amount of trial and error where the artist decides whether the parameters should be increased or decreased; this is only possible in the presence of near-instant visual feedback that reflects the effect of the parameter changes on the image. We propose a simple extension to our hybrid method to accommodate these workflows: consider an example scenario where the k-th target image in a series of target images $\tilde{\mathbf{t}}_{(k)}$ are produced by subjecting a starting image $\mathbf{t}$ to an increasingly wide blurring kernel. This operation is denoted by $\Psi_\sigma(\mathbf{t}) = G_\sigma * \mathbf{t}$, where G_σ is a zero-centered Gaussian, and for simplicity, the target images are produced via $\tilde{\mathbf{t}}_{(k)} = \Psi_k(\mathbf{t})$, with the initial condition of $\tilde{\mathbf{t}}_{(0)} = \mathbf{t}$. We note that many other transforms can also be substituted in the place of Ψ without loss of generality. We observe that such workflows create a series of images where each neighboring image pair shows only minute differences, i.e., for any positive non-zero k, $||\tilde{\mathbf{t}}_{(k+1)} - \tilde{\mathbf{t}}_{(k)}||_2$ remains small. As in these cases, we are required to propose many output images, we can take advantage of this favorable mathematical property by extending the pool of initial inversion networks with the optimized result of the previous frame by modifying Steps 3-5 of Algorithm 1 to add

$$\phi^{-1}_{(n+1)}(\tilde{\mathbf{t}}_k) = \underset{\mathbf{x}}{\text{argmin}} \left(||\phi(\mathbf{x}) - \tilde{\mathbf{t}}_{k-1}||_2 + \Gamma(\mathbf{x})\right). \tag{4}$$

Note that this does not require any extra computation as the result of Step 12 of the previous run can be stored and reused. Intuitively, this means that *both* the inversion network predictions and the prediction of the previous image are used as candidates for the optimization (whichever is better). This way, after the optimization step is finished, the improvements can be "carried over" to the next frame. This method we refer to as *reinitialization* and in Section 4, we show that it consistently improves the quality of our output images for such image sequences, even with a strict budget of 1-2 seconds per image.

	Initial guess		50 fun. evals		300 fun. evals		1500 fun. evals	
Input	Random	NN	Optimizer	Ours	Optimizer	Ours	Optimizer	Ours
Fig. 5, Row 1	41.93	5.94	33.81	**4.53**	9.42	**2.84**	5.62	**2.37**
Fig. 5, Row 2	78.45	32.72	68.55	**32.67**	40.24	**32.67**	40.21	**32.67**
Fig. 5, Row 4	35.37	18.68	30.88	**16.53**	17.29	**14.71**	16.98	**14.68**
Fig. 5, Row 7	41.65	22.42	38.10	**22.38**	26.30	**22.38**	26.24	**22.38**
Fig. 5, Row 8	29.04	19.82	26.79	**18.43**	22.93	**15.37**	22.93	**15.37**
Fig. 8, Row 2	23.78	12.79	20.31	**11.62**	8.27	**7.81**	8.26	**7.80**
Fig. 8, Row 3	21.60	9.09	16.54	**8.28**	6.24	**5.80**	6.19	**5.80**
Fig. 8, Row 8	29.58	9.74	22.69	**7.92**	6.63	**5.36**	6.63	**5.36**

Table 1: A comparison of the optimization approach (with random initialization) and our hybrid method (with "best of 9" NN initialization) on a variety of challenging global and local image editing operations in Fig. 5 and 8. The numbers indicate the RMSE of the outputs, and for reference, the first row showcases an input image that is reproducible by the shader.

		Image ID in sequence (i.e., k of $\tilde{\mathbf{t}}_{(k)}$)													
F. evals	Technique	0	10	20	30	40	50	60	70	80	90	100	110	120	Σ
100	No reinitialization	**1.93**	1.67	2.19	2.90	3.82	4.79	5.73	6.81	7.93	9.14	10.43	**11.55**	**12.99**	81.88
	Reinitialization	**1.93**	**1.34**	**1.88**	**2.54**	**3.34**	**4.30**	**5.30**	**6.38**	**7.50**	**8.69**	**9.93**	**11.55**	**12.99**	**77.67**
300	No reinitialization	**1.64**	1.47	2.07	2.80	3.70	4.62	5.70	6.75	7.86	9.00	10.21	**11.41**	**12.82**	80.05
	Reinitialization	**1.64**	**1.30**	**1.80**	**2.42**	**3.25**	**4.25**	**5.25**	**6.33**	**7.45**	**8.64**	**9.88**	**11.41**	**12.82**	**76.44**
600	No reinitialization	**1.57**	1.44	2.06	2.77	3.66	4.60	5.69	6.74	7.83	8.96	10.12	**11.41**	**12.80**	79.65
	Reinitialization	**1.57**	**1.29**	**1.80**	**2.49**	**3.33**	**4.20**	**5.18**	**6.27**	**7.38**	**8.58**	**9.81**	**11.41**	**12.80**	**76.11**

Table 2: Our proposed reinitialization technique consistently outperforms per-frame computation for the image sequence shown in Fig. 6. The numbers indicate the RMSE of the outputs.

4. Results

In this section, we discuss the properties of our inverse problem formulation (i.e., inferring a shader setup that produces a prescribed input image), followed by both a quantitative and qualitative evaluation of our proposed hybrid method against the optimization and inversion network solutions. We also show that our system supports a wide variety of image editing operations and can rapidly predict image sequences. To ensure clarity, we briefly revisit the three introduced methods:

- The **optimization** approach relies on minimizing (2) with Nelder and Mead's simplex method using a random initial guess, and implementing ϕ through a neural renderer,
- the **inversion network** refers to the "best of 9" inversion solution, i.e., $\mathbf{x} \approx \phi_{(i)}^{-1}(\tilde{\mathbf{t}})$ as shown in (3),
- our **hybrid method** is obtained by combining the two above approaches as described in Algorithm 1.

Furthermore, in Appendix A, we report the structure of the neural networks used to implement each individual $\phi_{(i)}^{-1}$ shown in Fig. 4, and compare our solution to a selection of local and global minimizers in Appendix B. At the end of this section, we also compare the total time taken to synthesize 1, 10, and 100 selected materials against a recent method for mass-scale material synthesis. Throughout this manuscript, all results were generated using a NVIDIA TITAN RTX GPU. The training set for the neural renderer is equivalent to the one used in Gaussian Material Synthesis [ZFWW18]. Our inversion networks are formulated as the adjoint of this neural renderer, and hence can be trained on the same dataset by swapping the inputs and outputs.

Inversion accuracy. Our inversion technique leads to an approximate solution within a few milliseconds, however, because the structure of the forward and inverse networks differ, the inversion operation remains imperfect, especially when presented with a target image that includes materials that are only approximately achievable. To demonstrate this effect, we have trained 9 different inversion networks to implement ϕ^{-1} and show that none of the proposed solutions are satisfactory as a final output for the global colorization case, and some may even predict results outside of the feasible domain (Fig. 4). Our goal with this experiment was to demonstrate that a solution containing only one inversion network generally produces unsatisfactory outputs, regardless of network structure. The reason for this is that the input images undergo a set of creative transforms by the artist and therefore differ significantly from the images contained within the training set. As a result, in most cases, an exact match is impossible to attain through the given principled shader. Due to the non-convex landscape of our principled shader, simply clamping back the parameters to the feasible domain may lead to undesirable results. One might consider using a final layer that passes the (to-be constrained) values through a suitable activation function (e.g., tanh). However, not all used quantities (e.g., volumetric absorption) are normalized, and tanh activations generally train less efficiently compared to ReLUs, especially when backpropagating gradients through many layers (we typically use 9 or more layers as discussed in Appendix A).

However, since we have an atypical problem where both the predicted images and the target image are available, we can inexpensively determine and choose the best prediction of a number of all of these inversion networks, leading to our "best-of-9" scheme. these predictions can be used to equip our optimizer with an initial guess, substantially improving its results. As each neural network consumes between 300MB and 1GB of video memory, we were able to keep all of them loaded during the entirety of the work session. We discuss the used architectures for all of these inversion networks in the Appendix and have included them in the supplementary materials as well.

Optimizer and hybrid solution accuracy. In Table 1, we compared our hybrid solution against the "best of 9" inversion network and optimization approaches and recorded the RMS error after 50, 300 and 1500 function evaluations (these roughly translate to 1, 6, and 30-second execution times) to showcase the early and late-stage performance of these methods. The table contains a selection of scenarios that we consider to be the most challenging and note that the outputs showed no meaningful change after 1500 function evaluations. Our hybrid method produced the lowest errors in each of our test cases, and surprisingly, the inversion network initialization not only provides a "headstart" for our method, but also improves the final quality of the output, thereby helping the optimizer to avoid local minima.

These results suggest that if real-time interaction is required, a two-stage system could be used where first, our inversion networks propose a reasonably accurate initial solution in a few milliseconds, and in the next stage, it is used as an initial guess by the optimizer and undergoes further refinement. In production rendering environments where the artist can typically afford to wait 20 seconds for a more accurate solution, we recommend using both stages. Furthermore, since both the input and the output images are available for the algorithm, the RMSE between the two can be compared. With a carefully chosen error threshold, this would result in a "best of both worlds" solution that only takes 20 seconds when necessary, and would execute in close to real time otherwise.

To validate the viability of our solutions, we also ran a global minimizer [WD97] with several different parameter choices and a generous allowance of 30 minutes of computation time for each; our hybrid method was often able to match (and in some cases, surpass) the quality offered by this solution (Appendix B, Table 3), further reinforcing how our inversion network initialization step helps avoid getting stuck in poor local minima. Note that the optimizer was unable to meaningfully improve the best prediction of the 9 inversion networks in Fig. 5, Row 7 – in this case, a better solution can be found by using the prediction of only the first neural network and passing it to the optimizer, improving the reported RMSE from 22.38 to 19.39 by using 300 function evaluations. This case is also the closest we have obtained to a failure case for our method, although we still consider it an acceptable result.

Example image editing operations. A typical workflow using our technique includes the artist choosing a source material and applying an appropriate image editing operation (Ψ) instead of engaging in a direct interaction with the principled shader. In collaboration with multiple artists, we selected a number of transforms that are likely to be relevant to a material-editing workflow and

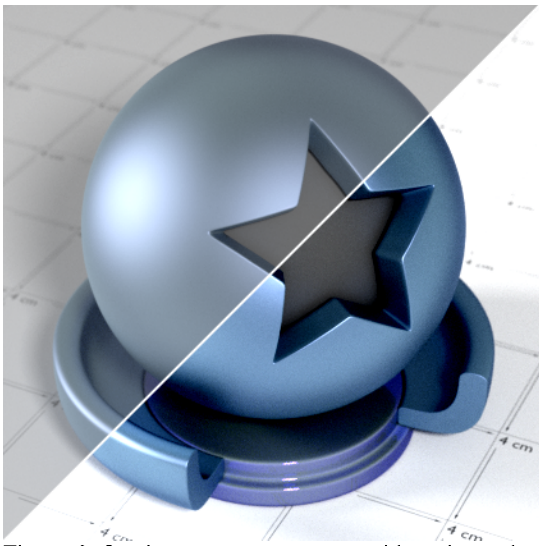

Figure 6: Our image sequence starts with an input that is achievable using our shader (upper left), where each animation frame slightly increases its black levels. The lower right region showcases the 300th frame of the animation.

cluster them into *global* (Fig. 5) and *local* (Fig. 8) operations: these include predominantly albedo-based changes, e.g., saturation and contrast enhancement, grayscale transform, changing the color balance or hue, and other image-based operations. e.g., image mixing, stitching and inpainting, and selective blurring of highlights. Other transforms should work as well (within limitations, see Sec. 5), as the system is trained independently of these operations.

Both the optimizer and our hybrid method were run for 1500 function evaluations to obtain the results showcased in these two figures. As these transformations come from a 2D raster editor and are not grounded in a physically based framework, a perfect match is often not possible, however, in each of these cases, our hybrid method proposed a solution of equivalent or better quality compared to the "best of 9" inversion network and the optimizer solutions.

Image sequence prediction. As our earlier results in Table 1 revealed that the global colorization techniques typically prove to be among the more difficult cases, we have created a challenging image sequence with an input image that is achievable with our shader, and subjected it to a slight black level increase over many frames (Fig. 6). Every image within this sequence is reproduced both with independent per-frame inference and our reinitialization technique with a strict time budget of 2, 6, and 12 seconds per image (100, 300, and 600 function evaluations). In Table 2, we show that this simple extension successfully exploits the advantageous mathematical properties of these workflows and consistently reduces the output error for the majority of the sequence, i.e., images 1-100. We also report the RMSE of images 101-120 for reference, which we refer to as the "converged" regime in which the target images stray further and further away from the feasible domain, and the proposed solution remains the same despite these changes. Even in these cases, our reinitialization technique performs no worse than the "no reinitialization" method, and because of its negligible additional cost, we consider it to be a strictly better solution.

Modeling and execution time. In Fig. 7, we have recorded the modeling times for 1, 10, and 100 similar materials using our method and compared them against Gaussian Material Synthesis [ZFWW18] (GMS), a learning-based technique for mass-scale material synthesis. We briefly describe the most important parameters of the task and refer the interested reader to this paper for more details. All timings are end-to-end, i.e., including all relevant user interaction and execution times. The task was to create a prescribed number n of materials that resemble (or match, in the case $n = 1$) a given target material. The novice and expert user timings were taken from the GMS paper and contain the time to created the materials by hand using Disney's "principled" shader [BS12]. The GMS timings contain scoring a material gallery by the user, computing suggestions, and selecting a desired material. Our timings contain the selection of a starting image from a set of images with random material parameters, the fixed cost of loading the 9 neural networks (5.5s), image processing operations by the user, as well as execution times.

If only one material is desired, our technique outperforms this previous work and nearly matches the efficiency of an expert user. When 10 similar materials are sought (1 base material and 9 variants), our proposed method was adapted to use the re-initialization technique and offers the best modeling times, outperforming both GMS and expert users. For 100 or more similar materials, both methods outperform experts, where GMS offers the best scaling solution – but note that in many practical scenarios, a scene is to be populated with many different materials of which only some are similar, which is equivalent to the 1- or 10-material cases we tested, leaving the advantage to GMS only in the specific use-case of mass-scale material synthesis. Also, discussions with artists suggest that our technique would often be used in different situations than GMS, namely editing an existing material vs. modeling from scratch. In this sense, it could also be used to fine-tune materials modeled with GMS.

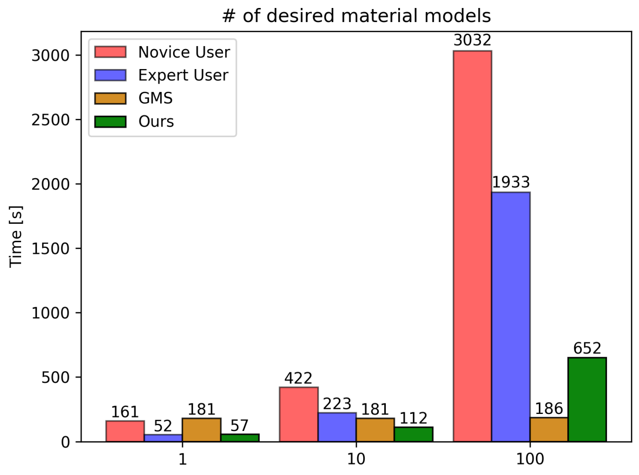

Figure 7: The recorded modeling times reveal that if at most a handful (i.e., 1-10) of target materials are sought, our technique offers a favorable entry point for novice users into the world of photorealistic material synthesis.

5. Limitations and Future Work

As demonstrated in Fig. 4, the results of ϕ^{-1} depend greatly on the performance of the encoder and decoder neural networks. As these methods enjoy significant research attention, we encourage further experiments in including these advances to improve them (e.g., architecture search [RMS*17], capsule networks [SFH17, HSF18] and skip connections [MSY16] among many other notable works) and adapting other neural network architectures to our problem that are more tailored to solve inverse problems [AKW*18, MEM19]. Even though our principled shader contains a set of parameters that are commonly used in the industry, there are other potential user interface options [KP10] that may lead to a difference in the modeling timings (Fig. 7). Furthermore, strongly localized edits, e.g., blurring a small part of a specular highlight typically introduces drastic changes within only a small subset of the image and represent only a small fraction of the RMSE calculations and thus may not get proper prioritization from the optimizer. To alleviate this, the relative importance of different regions may also be controlled via weighted masks to emphasize these edits, making these edited regions "score higher" in the error metric, offering the user more granular artistic control. In specialized cases, our reinitialization technique may prove to be useful for single images by using the parameter set used to produce $\mathbf{t}$ as an initial guess for $\tilde{\mathbf{t}}$. In-scene editing still remains the key advantage of BRDF relighting techniques.

We also note that our learning technique assumes an input shader of dimensionality m and a renderer that is able to produce images of the materials that it encodes. In this work, our principled shader was meant to demonstrate the utility of this approach by showcasing intuitive workflows with the most commonly used BSDFs. However, this method needs not to be restricted to our principled BSDF, and is also expected to perform well on a rich selection of more specialized material models including thin-film interference [Dia91, IWR*15], fluorescence [WTP01] birefringence [WW08], microfacet models [HHdD16] layered materials [Bel18, ZJ18], materials with Fresnel effects, and more.

Our method relies on using the same scene and lighting setup for the editing session, as changing these would require retraining the network. We do not consider this a major limitation as we carefully chose a material test scene that is widely used by material editing artists working in the industry. The same scene was also used in GMS [ZFWW18]. Even though starting the editing process from an arbitrary image is not possible with this proposed system, we conjecture that it can be extended to work with real photographs as inputs – this can likely be achieved through the presence of a mechanism for detecting and lining up the specular highlights with the image of our material editing scene [FJL*16].

It would be interesting to investigate gradients for the loss function. This is challenging since these neural networks are typically very sensitive to the complexity of the loss function and may become more difficult to train properly.

For error measurements, we used RMSE as it is the standard

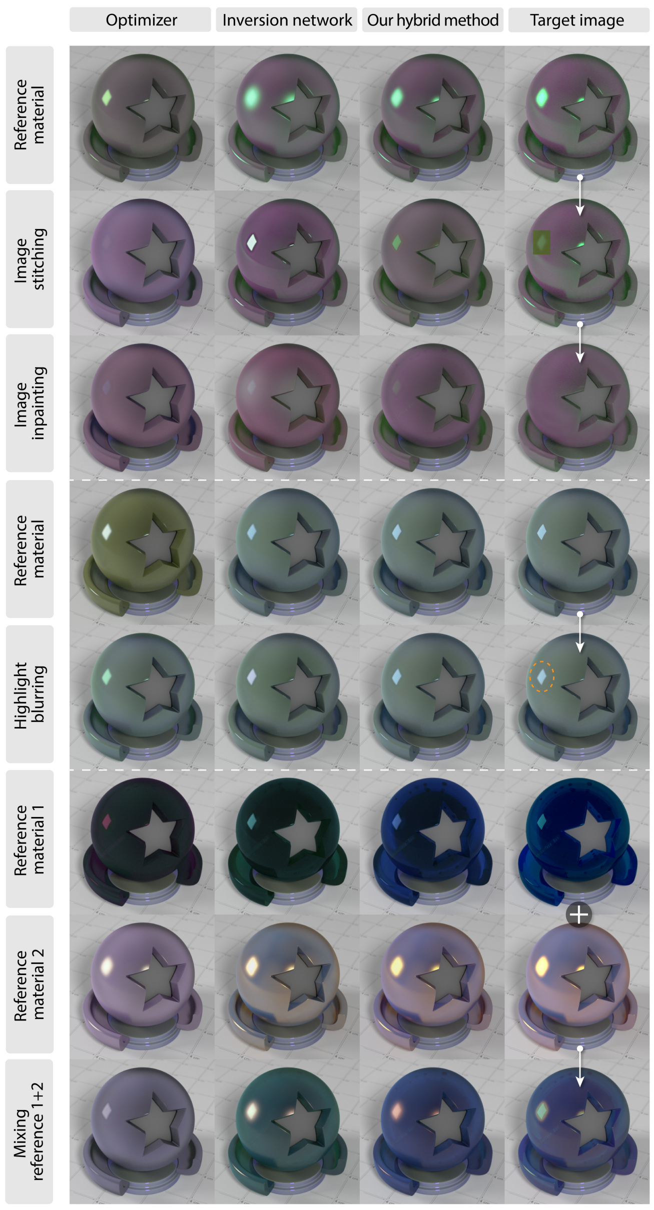

Figure 8: Results for three techniques on local image editing operations and image mixing. Please see Fig. 5 for an explanation of reference material and arrows.

way of measuring differences in BRDF modeling [DJ18]. There are specialized cases, e.g., noise and blurring among other examples, that would likely require non-standard or perceptual image quality metrics. Regardless, we have tried measuring the PSNR and produced per-channel greyscale images to record the SSIM and have not found meaningful differences to RMSE in our test cases.

6. Conclusions

We have presented a hybrid technique to empower novice users and artists without expertise in photorealistic rendering to create sophisticated material models by applying image editing operations to a source image. This allows them to reuse their image editing knowledge and apply it to material synthesis. The resulting images are typically not achievable through photorealistic rendering, however, in many cases, solutions be found that are close to the desired output. Our learning-based technique is able to take such an edited image and propose a photorealistic material setup that produces a similar output, and provides high-quality results even in the presence of poorly-edited images. Our proposed method produces a reasonable initial guess and uses a neural network-augmented optimizer to fine-tune the parameters until the target image is matched as closely as possible. This hybrid method is simple, robust, and its computation time is within 30 seconds for every test case showcased throughout this paper. This low computation time is beneficial especially in the early phases of the material design process where a rapid iteration over a variety of competing ideas is an important requirement (Fig. 9). Our key insights can be summarized as follows:

- Normally, using an input image that was generated by a principled shader is not useful given that the user has to generate this image themselves with a known parameter setup. However, our main idea is that the user can subject this image to raster editing operations and "pretend" that this input is achievable, and reliably infer a shader setup to mimic it.
- Our neural networks can be combined with optimizers both *directly*, i.e., by using an optimizer that invokes a neural renderer at every function evaluation step to speed up the convergence and *indirectly* by using a set of neural networks network to endow the optimizer with a reasonable initial guess (steps ❸ and ❹ in Fig. 2).
- Our inversion problem is quite difficult to solve reliably – the reason why this happens is that the edited images often stray far away from the samples contained within the training set, and it is not feasible to train them on all possible artistic edits. To alleviate this, our system combines multiple, otherwise unreliable neural network predictions with an optimizer to be able to match these inputs.

Furthermore, we proposed a simple extension to support predicting image sequences with a strict time budget of 1-2 seconds and believe this method will offer an appealing entry point for novices into world of photorealistic material modeling.

Acknowledgments

We would like to thank Reynante Martinez for providing us the geometry and some of the materials for the Paradigm (Fig. 1) and Genesis scenes (Fig. 3), ianofshields for the Liquify scene that served as a basis for Fig. 9, Robin Marin for the material test scene, Andrew Price and Gábor Mészáros for their help with geometry modeling, Felícia Zsolnai-Fehér for her help improving our figures, Christian Freude, David Ha, Philipp Erler and Adam Celarek for their useful comments. We also thank the anonymous reviewers for their help improving this manuscript and NVIDIA for providing the hardware to train our neural networks. This work was partially funded by Austrian Science Fund (FWF), project number P27974.

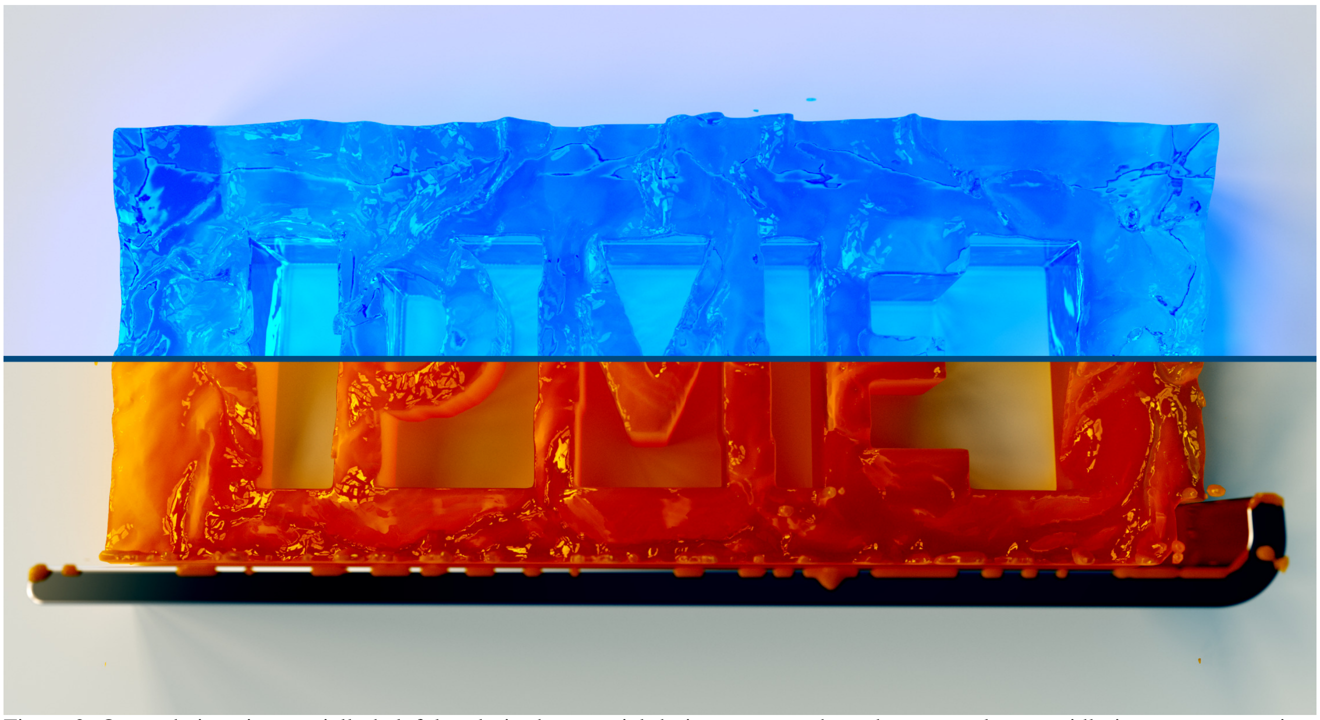
Figure 9: Our technique is especially helpful early in the material design process where the user seeks to rapidly iterate over a variety of possible artistic effects. Both material types were designed using the material test scene (see Fig. 5) and then transferred to the scene in this figure. We also demonstrate the method in our supplementary video.

Appendix A: Neural network architectures

Below, we describe the neural network architectures we used to implement $\phi_{(i)}^{-1}$. The Conv2D notation represents a 2D convolutional layer with the appropriate *number of filters*, *spatial kernel sizes* and *strides*, where FC represents a dense, fully-connected layer with a prescribed number of *neurons* and *dropout probability*.

1. 2x{Conv2D(32,3,1), MaxPool(2,2)} – 1x{Conv2D(64,3,1), MaxPool(2,2)} – 2x{Conv2D(128,3,1), MaxPool(2,2)} – 2x{FC(1000, 0.1)} - FC(m, 0.0)
2. 2x{Conv2D(32,3,1), MaxPool(2,2)} – 2x{FC(1000, 0.1)} - FC(m, 0.0)
3. 2x{Conv2D(32,3,1), MaxPool(2,2)} – 2x{FC(1000, 0.5)} - FC(m, 0.0)
4. 2x{Conv2D(32,3,1), MaxPool(2,2)} – 1x{Conv2D(64,3,1), MaxPool(2,2)} – 2x{Conv2D(128,3,1), MaxPool(2,2)} – 2x{FC(3000, 0.5)} - FC(m, 0.0)
5. 2x{Conv2D(32,3,1), MaxPool(2,2)} – 1x{Conv2D(64,3,1), MaxPool(2,2)} – 2x{Conv2D(128,3,1), MaxPool(2,2)} – 2x{FC(3000, 0.0)} - FC(m, 0.0)
6. 2x{Conv2D(32,3,1), MaxPool(2,2)} – 2x{FC(1000, 0.0)} - FC(m, 0.0)
7. 2x{Conv2D(32,3,1), MaxPool(2,2)} – 2x{FC(1000, 0.0)} - FC(m, 0.0)
8. 2x{Conv2D(32,3,1), MaxPool(2,2)} – 2x{FC(100, 0.0)} - FC(m, 0.0)
9. 2x{Conv2D(32,3,1), MaxPool(2,2)} – 2x{FC(1000, 0.0)} - FC(m, 0.0)

Neural networks 6,7 and 9 are isomorphic and were run for a different number of epochs to test the effect of overfitting later in the training process, and therefore offer differing validation losses. The implementation of ϕ is equivalent to the one used in Zsolnai-Fehér et al.'s work [ZFWW18].

Appendix B: Comparison of optimizers

In Table 3, we have benchmarked several optimizers, i.e., L-BFGS-B [BLNZ95], SLSQP [Kra94], the Conjugate Gradient method [HS52] and found Nelder and Mead's simplex-based self-adapting optimizer [NM65] to be the overall best choice for our global and local image-editing operations. For reference, we also ran Basin-hopping [WD97], a global minimizer with a variety of parameter choices and a generous allowance of 30 minutes of execution time for each test case. This method is useful for challenging non-linear optimization problems with high-dimensional search spaces. Note that when being run for long enough, this technique is less sensitive to initialization due to the fact that it performs many quick runs from different starting points, and hence, we report one result for both initialization techniques. The cells in the intersection of "Nelder-Mead" and "NN" denote our proposed hybrid method, which was often able to match, and in some cases, outperform this global minimization technique.

Input	Init. type	Init. RMSE	Nelder-Mead	L-BFGS-B	SLSQP	CG	Basin-hopping
Fig. 5, Row 1	Rand	41.93	5.62	20.47	17.96	5.24	**2.01**
Fig. 5, Row 1	NN	5.94	2.37	5.84	5.94	5.94	
Fig. 5, Row 2	Rand	78.45	40.21	78.45	78.45	78.45	**32.67**
Fig. 5, Row 2	NN	32.72	**32.67**	32.72	32.72	32.72	
Fig. 5, Row 4	Rand	35.37	16.98	28.84	35.37	34.99	14.72
Fig. 5, Row 4	NN	18.68	**14.68**	15.33	18.18	15.90	
Fig. 5, Row 7	Rand	41.65	26.24	41.65	41.65	41.65	**22.38**
Fig. 5, Row 7	NN	22.42	**22.38**	22.42	22.42	22.42	
Fig. 5, Row 8	Rand	29.04	22.93	29.04	26.71	28.21	15.69
Fig. 5, Row 8	NN	19.82	**15.37**	19.82	28.87	19.82	
Fig. 8, Row 2	Rand	23.78	8.26	23.78	23.78	21.75	**7.63**
Fig. 8, Row 2	NN	12.79	7.80	12.79	12.79	12.79	
Fig. 8, Row 3	Rand	21.60	6.19	21.60	21.60	20.83	5.86
Fig. 8, Row 3	NN	9.09	**5.80**	9.09	9.09	9.09	
Fig. 8, Row 8	Rand	29.58	6.63	29.58	29.58	29.58	**5.07**
Fig. 8, Row 8	NN	9.74	5.36	9.61	9.61	9.68	

Table 3: A comparison of a set of classical optimization techniques revealed that when using Nelder and Mead's simplex-based optimizer with our "best of 9" inversion network initialization, we can often match, and in some cases, outperform the results of Basin-hopping, a global minimizer. In the interest of readability, we have marked the cases where the optimizers were unable to improve upon the initial guess with red. For reference, the first two rows showcase an input image that is reproducible by the shader.

References

[AAL16] Aittala M., Aila T., Lehtinen J.: Reflectance modeling by neural texture synthesis. *ACM Transactions on Graphics 35*, 4 (2016), 65. 2

[AKW*18] Ardizzone L., Kruse J., Wirkert S., Rahner D., Pellegrini E. W., Klessen R. S., Maier-Hein L., Rother C., Köthe U.: Analyzing inverse problems with invertible neural networks. *arXiv preprint arXiv:1808.04730* (2018). 9

[ATDP11] An X., Tong X., Denning J. D., Pellacini F.: Appwarp: retargeting measured materials by appearance-space warping. *ACM Trans. Graph. 30*, 6 (2011), 147. URL: https://doi.org/10.1145/2070781.2024181, doi:10.1145/2070781.2024181. 2

[AWL*15] Aittala M., Weyrich T., Lehtinen J., et al.: Two-shot svbrdf capture for stationary materials. *ACM Transactions on Graphics 34*, 4 (2015), 110–1. 2

[BAEDR08] Ben-Artzi A., Egan K., Durand F., Ramamoorthi R.: A precomputed polynomial representation for interactive brdf editing with global illumination. *ACM Transactions on Graphics (TOG) 27*, 2 (2008), 13. 2

[BAOR06] Ben-Artzi A., Overbeck R., Ramamoorthi R.: Real-time brdf editing in complex lighting. *ACM Transactions on Graphics 25*, 3 (2006), 945–954. 2

[Bel18] Belcour L.: Efficient Rendering of Layered Materials using an Atomic Decomposition with Statistical Operators. *ACM Transactions on Graphics 37*, 4 (2018), 1. URL: https://hal.archives-ouvertes.fr/hal-01785457, doi:10.1145/3197517.3201289. 10

[BLNZ95] Byrd R. H., Lu P., Nocedal J., Zhu C.: A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing 16*, 5 (1995), 1190–1208. 11

[Bot10] Bottou L.: Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*. Springer, 2010, pp. 177–186. 2

[BS12] Burley B., Studios W. D. A.: Physically-based shading at disney. In *ACM SIGGRAPH* (2012), vol. 2012, pp. 1–7. 2, 9

[BZVL17] Bello I., Zoph B., Vasudevan V., Le Q. V.: Neural optimizer search with reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70* (2017), JMLR. org, pp. 459–468. 3

[CLZ*18] Che C., Luan F., Zhao S., Bala K., Gkioulekas I.: Inverse transport networks. *CoRR abs/1809.10820* (2018). URL: http://arxiv.org/abs/1809.10820, arXiv:1809.10820. 2

[CPWAP08] Cheslack-Postava E., Wang R., Akerlund O., Pellacini F.: Fast, realistic lighting and material design using nonlinear cut approximation. *ACM Transactions on Graphics 27*, 5 (2008), 128. 2

[DAD*18] Deschaintre V., Aittala M., Durand F., Drettakis G., Bousseau A.: Single-image svbrdf capture with a rendering-aware deep network. *ACM Transactions on Graphics (TOG) 37*, 4 (2018), 128. 2

[Dia91] Dias M. L.: Ray tracing interference color. *IEEE Computer Graphics and Applications*, 2 (1991), 54–60. 10

[DJ18] Dupuy J., Jakob W.: An adaptive parameterization for efficient material acquisition and rendering. *Transactions on Graphics (Proceedings of SIGGRAPH Asia)* (Dec. 2018). 2, 10

[DTPG11] Dong Y., Tong X., Pellacini F., Guo B.: Appgen: interactive material modeling from a single image. *ACM Trans. Graph. 30*, 6 (2011), 146. URL: https://doi.org/10.1145/2070781.2024180, doi:10.1145/2070781.2024180. 2

[EMH18] Elsken T., Metzen J. H., Hutter F.: Neural architecture search: A survey. *arXiv preprint arXiv:1808.05377* (2018). 3

[FJL*16] Fiser J., Jamriska O., Lukác M., Shechtman E., Asente P., Lu J., Sýkora D.: Stylit: illumination-guided example-based stylization of 3d renderings. *ACM Trans. Graph. 35*, 4 (2016), 92:1–92:11. URL: https://doi.org/10.1145/2897824.2925948, doi:10.1145/2897824.2925948. 10

[GGG*16] Guarnera D., Guarnera G. C., Ghosh A., Denk C., Glencross M.: BRDF representation and acquisition. *Comput.*

Graph. Forum 35, 2 (2016), 625–650. URL: https://doi.org/10.1111/cgf.12867, doi:10.1111/cgf.12867. 2

[GLD*19] Gao D., Li X., Dong Y., Peers P., Xu K., Tong X.: Deep inverse rendering for high-resolution svbrdf estimation from an arbitrary number of images. *ACM Transactions on Graphics (TOG) 38*, 4 (2019), 134. 3

[Goh17] Goh G.: Why momentum really works. *Distill 2*, 4 (2017), e6. 3

[GRR*17] Georgoulis S., Rematas K., Ritschel T., Gavves E., Fritz M., Van Gool L., Tuytelaars T.: Reflectance and natural illumination from single-material specular objects using deep learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence 40*, 8 (2017), 1932–1947. 2

[HDR19] Hu Y., Dorsey J., Rushmeier H. E.: A novel framework for inverse procedural texture modeling. *ACM Trans. Graph. 38*, 6 (2019), 186:1–186:14. URL: https://doi.org/10.1145/3355089.3356516, doi:10.1145/3355089.3356516. 3

[HHdD16] Heitz E., Hanika J., d'Eon E., Dachsbacher C.: Multiple-scattering microfacet bsdfs with the smith model. *ACM Transactions on Graphics (TOG) 35*, 4 (2016), 58. 10

[HR13] Hašan M., Ramamoorthi R.: Interactive albedo editing in path-traced volumetric materials. *ACM Transactions on Graphics (TOG) 32*, 2 (2013), 11. 2

[HS52] Hestenes M. R., Stiefel E.: Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards 49*, 1 (1952), 409–435. 11

[HSF18] Hinton G. E., Sabour S., Frosst N.: Matrix capsules with em routing. In *6th international conference on learning representations, ICLR* (2018), pp. 1–15. 9

[IWR*15] Ikeda S., Watanabe S., Raytchev B., Tamaki T., Kaneda K.: Spectral rendering of interference phenomena caused by multilayer films under global illumination environment. *ITE Transactions on Media Technology and Applications 3*, 1 (2015), 76–84. 10

[KB14] Kingma D., Ba J.: Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014). 3

[KCW*18] Kang K., Chen Z., Wang J., Zhou K., Wu H.: Efficient reflectance capture using an autoencoder. *ACM Trans. Graph. 37*, 4 (2018), 127:1–127:10. URL: https://doi.org/10.1145/3197517.3201279, doi:10.1145/3197517.3201279. 3

[KP10] Kerr W. B., Pellacini F.: Toward evaluating material design interface paradigms for novice users. *ACM Trans. Graph. 29*, 4 (2010), 35:1–35:10. URL: https://doi.org/10.1145/1778765.1778772, doi:10.1145/1778765.1778772. 9

[Kra94] Kraft D.: Algorithm 733: Tomp–fortran modules for optimal control calculations. *ACM Transactions on Mathematical Software (TOMS) 20*, 3 (1994), 262–281. 11

[LCY*17] Liu G., Ceylan D., Yumer E., Yang J., Lien J.: Material editing using a physically based rendering network. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017* (2017), IEEE Computer Society, pp. 2280–2288. URL: https://doi.org/10.1109/ICCV.2017.248, doi:10.1109/ICCV.2017.248. 2

[LDPT17] Li X., Dong Y., Peers P., Tong X.: Modeling surface appearance from a single photograph using self-augmented convolutional neural networks. *ACM Transactions on Graphics (TOG) 36*, 4 (2017), 45. 2

[LHJ19] Loubet G., Holzschuch N., Jakob W.: Reparameterizing discontinuous integrands for differentiable rendering. *ACM Trans. Graph. 38*, 6 (2019), 228:1–228:14. URL: https://doi.org/10.1145/3355089.3356510, doi:10.1145/3355089.3356510. 2

[LMS*19] Lagunas M., Malpica S., Serrano A., Garces E., Gutierrez D., Masia B.: A similarity measure for material appearance. *ACM Transactions on Graphics (SIGGRAPH 2019) 38*, 4 (2019). 2

[LSC18] Li Z., Sunkavalli K., Chandraker M.: Materials for masses: Svbrdf acquisition with a single mobile phone image. In *Proceedings of the European Conference on Computer Vision (ECCV)* (2018), pp. 72–87. 2

[Mat03] Matusik W.: *A data-driven reflectance model*. PhD thesis, Massachusetts Institute of Technology, 2003. 2

[MEM19] Mataev G., Elad M., Milanfar P.: Deepred: Deep image prior powered by red, 2019. arXiv:1903.10176. 9

[MG98] Marschner S. R., Greenberg D. P.: *Inverse rendering for computer graphics*. Citeseer, 1998. 2

[MSY16] Mao X., Shen C., Yang Y.-B.: Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In *Advances in neural information processing systems* (2016), pp. 2802–2810. 9

[NH92] Nowlan S. J., Hinton G. E.: Simplifying neural networks by soft weight-sharing. *Neural Computation 4*, 4 (1992), 473–493. 5

[NM65] Nelder J. A., Mead R.: A simplex method for function minimization. *The computer journal 7*, 4 (1965), 308–313. 4, 11

[NRH04] Ng R., Ramamoorthi R., Hanrahan P.: Triple product wavelet integrals for all-frequency relighting. *ACM Transactions on Graphics (TOG) 23*, 3 (2004), 477–487. 2

[NSR17] Nalbach O., Seidel H.-P., Ritschel T.: Practical capture and reproduction of phosphorescent appearance. *Computer Graphics Forum 36*, 2 (2017), 409–420. 2

[NSRS13] Nguyen C. H., Scherzer D., Ritschel T., Seidel H.-P.: Material editing in complex scenes by surface light field manipulation and reflectance optimization. *Computer Graphics Forum 32*, 2pt2 (2013), 185–194. 2

[PdMJ14] Papas M., de Mesa K., Jensen H. W.: A physically-based bsdf for modeling the appearance of paper. *Computer Graphics Forum 33*, 4 (2014), 133–142. 2

[PL07] Pellacini F., Lawrence J.: Appwand: editing measured materials using appearance-driven optimization. *ACM Trans. Graph. 26*, 3 (2007), 54. URL: https://doi.org/10.1145/1276377.1276444, doi:10.1145/1276377.1276444. 2

[PRJ*13] Papas M., Regg C., Jarosz W., Bickel B., Jackson P., Matusik W., Marschner S., Gross M.: Fabricating translucent materials using continuous pigment mixtures. *ACM Transactions on Graphics (TOG) 32*, 4 (2013), 146. 2

[RCP14] Renzo F. D., Calabrese C., Pellacini F.: Appim: linear spaces for image-based appearance editing. *ACM Trans. Graph. 33*, 6 (2014), 194:1–194:9. URL: https://doi.org/10.1145/2661229.2661282, doi:10.1145/2661229.2661282. 2

[RH01] Ramamoorthi R., Hanrahan P.: A signal-processing framework for inverse rendering. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (2001), ACM, pp. 117–128. 2

[RJGW19] Rainer G., Jakob W., Ghosh A., Weyrich T.: Neural btf compression and interpolation. *Computer Graphics Forum (Proceedings of Eurographics) 38*, 2 (Mar. 2019). 2

[RM51] Robbins H., Monro S.: A stochastic approximation method. *The annals of mathematical statistics* (1951), 400–407. 2

[RMS*17] Real E., Moore S., Selle A., Saxena S., Suematsu Y. L., Tan J., Le Q., Kurakin A.: Large-scale evolution of image classifiers. *arXiv preprint arXiv:1703.01041* (2017). 9

[RSB*02] Röder B., Stock O., Bien S., Neville H., Rösler F.: Speech processing activates visual cortex in congenitally blind humans. *European Journal of Neuroscience 16*, 5 (2002), 930–936. 2

[SFH17] SABOUR S., FROSST N., HINTON G. E.: Dynamic routing between capsules. In *Advances in Neural Information Processing Systems* (2017), pp. 3856–3866. 9

[SGM*16] SERRANO A., GUTIERREZ D., MYSZKOWSKI K., SEIDEL H.-P., MASIA B.: An intuitive control space for material appearance. *ACM Transactions on Graphics 35*, 6 (2016), 186. 2

[SHK*14] SRIVASTAVA N., HINTON G. E., KRIZHEVSKY A., SUTSKEVER I., SALAKHUTDINOV R.: Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research 15*, 1 (2014), 1929–1958. 5

[SJR18] SUN T., JENSEN H. W., RAMAMOORTHI R.: Connecting measured brdfs to analytic brdfs by data-driven diffuse-specular separation. *ACM Transactions on Graphics (TOG) 37*, 6 (2018), 1–15. 2

[SKWW17] SZTRAJMAN A., KRIVÁNEK J., WILKIE A., WEYRICH T.: Image-based remapping of material appearance. In *MAM@EGSR 2017: Eurographics Workshop on Material Appearance Modeling, Helsinki, Finland, 18 June 2017, Held in conjunction with The 28th Eurographics Symposium on Rendering* (2017), Klein R., Rushmeier H. E., (Eds.), Eurographics Association, pp. 5–8. URL: https://doi.org/10.2312/mam.20171323, doi:10.2312/mam.20171323. 2

[SMDH13] SUTSKEVER I., MARTENS J., DAHL G., HINTON G.: On the importance of initialization and momentum in deep learning. In *International conference on machine learning* (2013), pp. 1139–1147. 3

[SNM*13] SCHMIDT T.-W., NOVAK J., MENG J., KAPLANYAN A. S., REINER T., NOWROUZEZAHRAI D., DACHSBACHER C.: Path-space manipulation of physically-based light transport. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2013) 32*, 4 (Aug. 2013). 2

[SSN18] SOLER C., SUBR K., NOWROUZEZAHRAI D.: A versatile parameterization for measured material manifolds. *Computer Graphics Forum 37*, 2 (2018), 135–144. 2

[STPP09] SONG Y., TONG X., PELLACINI F., PEERS P.: Subedit: a representation for editing measured heterogeneous subsurface scattering. *ACM Transactions on Graphics (TOG) 28*, 3 (2009), 31. 2

[SZC*07] SUN X., ZHOU K., CHEN Y., LIN S., SHI J., GUO B.: Interactive relighting with dynamic brdfs. *ACM Transactions on Graphics 26*, 3 (2007), 27. 2

[WAA*00] WOOD D. N., AZUMA D. I., ALDINGER K., CURLESS B., DUCHAMP T., SALESIN D., STUETZLE W.: Surface light fields for 3d photography. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2000, New Orleans, LA, USA, July 23-28, 2000* (2000), Brown J. R., Akeley K., (Eds.), ACM, pp. 287–296. URL: https://doi.org/10.1145/344779.344925, doi:10.1145/344779.344925. 2

[WCPL*08] WANG R., CHESLACK-POSTAVA E., LUEBKE D., CHEN Q., HUA W., PENG Q., BAO H.: Real-time editing and relighting of homogeneous translucent materials. *The Visual Computer 24*, 7-9 (2008), 565–575. 2

[WD97] WALES D. J., DOYE J. P.: Global optimization by basin-hopping and the lowest energy structures of lennard-jones clusters containing up to 110 atoms. *The Journal of Physical Chemistry A 101*, 28 (1997), 5111–5116. 8, 11

[WdBKK15] WEINMANN M., DEN BROK D., KRUMPEN S., KLEIN R.: Appearance capture and modeling. In *SIGGRAPH Asia 2015 Courses, Kobe, Japan, November 2-6, 2015* (2015), ACM, p. 4:1. URL: https://doi.org/10.1145/2818143.2835226, doi:10.1145/2818143.2835226. 2

[WDR13] WU H., DORSEY J., RUSHMEIER H. E.: Inverse bi-scale material design. *ACM Trans. Graph. 32*, 6 (2013), 163:1–163:10. URL: https://doi.org/10.1145/2508363.2508394, doi:10.1145/2508363.2508394. 2

[Whi89] WHITE R.: Visual thinking in the ice age. *Scientific American 261*, 1 (1989), 92–99. 2

[WTL04] WANG R., TRAN J., LUEBKE D. P.: All-frequency relighting of non-diffuse objects using separable brdf approximation. In *Rendering Techniques* (2004), pp. 345–354. 2

[WTP01] WILKIE A., TOBLER R. F., PURGATHOFER W.: Combined rendering of polarization and fluorescence effects. In *Rendering Techniques 2001*. Springer, 2001, pp. 197–204. 10

[WW08] WEIDLICH A., WILKIE A.: Realistic rendering of birefringency in uniaxial crystals. *ACM Transactions on Graphics (TOG) 27*, 1 (2008), 6. 10

[ZFWW18] ZSOLNAI-FEHÉR K., WONKA P., WIMMER M.: Gaussian material synthesis. *ACM Transactions on Graphics (Proc. SIGGRAPH)* (2018). 2, 3, 4, 7, 8, 10, 11

[ZH05] ZOU H., HASTIE T.: Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 67*, 2 (2005), 301–320. 5

[ZJ18] ZELTNER T., JAKOB W.: The layer laboratory: a calculus for additive and subtractive composition of anisotropic surface reflectance. *Transactions on Graphics (Proceedings of SIGGRAPH) 37*, 4 (July 2018), 74:1–74:14. doi:10.1145/3197517.3201321. 10

[ZKSE16] ZHU J.-Y., KRÄHENBÜHL P., SHECHTMAN E., EFROS A. A.: Generative visual manipulation on the natural image manifold. In *European Conference on Computer Vision* (2016), Springer, pp. 597–613. 3

[ZL16] ZOPH B., LE Q. V.: Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578* (2016). 3