FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Photorealistic Material Learning and Synthesis

DISSERTATION

submitted in partial fulfillment of the requirements for the degree of

Doktor der Technischen Wissenschaften

by

MSc. Károly Zsolnai-Fehér
Registration Number 01229739

to the Faculty of Informatics

at the TU Wien

Advisor: Michael Wimmer, Assoc. Prof. Dipl.-Ing. Dipl.-Ing. Dr.techn.

The dissertation has been reviewed by:

Torsten Möller	Derek Nowrouzezahrai

Vienna, 26th October, 2019

Károly Zsolnai-Fehér

Erklärung zur Verfassung der Arbeit

MSc. Károly Zsolnai-Fehér

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 26. Oktober 2019

Károly Zsolnai-Fehér

Danksagung

Zunächst möchte ich mich bei Michael Wimmer, meinem Betreuer, bedanken, der mir die Möglichkeit gegeben hat in seiner Forschungsgruppe zu promovieren. Während unserer gesamten Zusammenarbeit war er immer offen für tiefgehende mathematische Gespräche und ermöglichte mir einen Zeitplan in dem ich meine beste Arbeit leisten konnte. Dass ich meine Themen völlig frei wählen konnte, ist das Beste was sich ein junger Forscher wünschen kann. Ich danke Peter Wonka dafür, dass er mir in der zweiten Hälfte meiner Doktorarbeit geholfen hat und nebst vielen wichtigen Beiträgen oft seine unschätzbar wertvollen Erkenntnisse einbrachte und meine Mathematik häufig zerlegte um sie besser zusammen zu fügen.

Ich bin Thomas Auzinger, Hiroyuki Sakai und Christian Freude für unsere vielen fruchtbaren Gespräche dankbar, denn ich habe viel von Ihnen in der unübertroffen Atmosphäre an unserem Institut gelernt. Es war eine große Freude, mit so vielen glücklichen Menschen zusammen zu arbeiten. Ich danke auch Till von Ahnen für seine Hilfe bei der Übersetzung des Vorwortes und der Danksagungen dieser Arbeit ins Deutsche.

Forschung ist die Untersuchung des Scheiterns. Genauer gesagt, ist Forschung die Lehre des Erwerbs neuer Erkenntnisse durch Scheitern. Daher scheitert ein schlechter Forscher zu 100% in seinen Versuchen, während ein guter nur zu 99% scheitert. Daher ist das was Sie hier lesen, wie in vielen anderen Arbeiten, nur 1% der Forschungsarbeit. Ich möchte Felícia, meiner Frau, danken, dass sie mich motiviert hat, mich vor Ablenkungen bewahrt hat und mir die Sonne in mein Leben gebracht hat um viele dieser Misserfolge zu überstehen.

Abschließend möchte ich meiner Familie danken die mich trotz ihrer schwierigen Leben immer wieder ermutigt hat mehr zu lernen und mich bei meinen Bestreben unterstützt hat. Mein Vater lebte und arbeitete fast sein ganzes Leben im Ausland, um finanzielle Schwierigkeiten der Familie zu lindern. Während meine Mutter uns drei Kinder neben einem Vollzeitjob aufgezogen hat. Ohne euch wäre all dies nicht möglich gewesen.

Mein Ziel ist es, schöne wissenschaftliche Abhandlungen zu schreiben, die ein Feuer in Menschen entfachen. Vielen Dank an euch alle, dass ihr mir bei dieser Aufgabe geholfen habt.

Acknowledgements

First, I would like send a big thank you to Michael Wimmer, my advisor, for giving me a chance to pursue a PhD in his research group. Throughout our work together, he was always open for deep mathematical conversations and made me a schedule in which I could perform my best work. I was also able to choose my topics with complete freedom, which is everything a young researcher could ask for. I also thank Peter Wonka for helping me during the second half of my PhD - among many other important contributions, his insights were invaluable and he was frequently able to pick apart my mathematics and improve it further.

I am grateful for Thomas Auzinger, Hiroyuki Sakai and Christian Freude for our many fruitful discussions - I have learned a lot from you and the atmosphere at our institute has been second to none. It's been such a joy to work with and among so many happy people. I also thank Till von Ahnen for his help translating the abstract and the acknowledgements sections of this thesis to German.

Research is the study of failure. More precisely, research is the study of obtaining new knowledge through failure. A bad researcher fails 100% of the time, while a good one fails only 99% of the time. Hence, what you see written here (and in most papers) is only 1% of the work that has been done. I would like to thank Felícia, my wife, for providing motivation, shielding me from distractions, and bringing sunshine to my life to endure through many of these failures.

Finally, I would like express my gratitude to my family, who, despite their difficult lives, always encouraged me to study more and supported me throughout my endeavors. My father lived and worked abroad for most his life to be able to support our family through financial hardships, while my mother raised the three of us while attending to a full-time job. None of this would have been possible without you.

My objective is to write beautiful papers that light a fire in people. Thank you very much to all of you for helping me with this quest.

Kurzfassung

Die Simulation von Licht ist der Industriestandard, um überzeugende fotorealistische Bilder zu erzeugen und wird häufig bei der Erstellung von Animationsfilmen, Computeranimationen, in Architektur-/ Medizinvisualisierung und vielen weiteren nennenswerten Anwendungen eingesetzt. Da diese Techniken die Interaktionen zwischen Millionen von Lichtstrahlen in virtuellen Szenen simulieren ist das Endergebniss sehr stark von der Qualität der benutzten Materialien und Objektgeometrien in der Szene abhängig. In dieser Dissertation behandeln wir zwei Kernprobleme bezüglich der photorealistischen Materialsynthese.

Erstens, das Erstellen von überzeugenden photorealistischen Materialien. Das Erlernen dieser Fähigkeit erfordert jahrelange Erfahrung eines geschulten Künstlers und selbst dann ist ein nicht trivialer Zeitaufwand pro Material seitens des Künstlers gefüllt mit Ausprobieren und Nachbessern. Wir schlagen zwei lernbasierte Methoden vor, welche es unerfahrenen Nutzern ermöglichen, einfach und schnell photorealistische Materialien zu synthetisieren indem die Präferenzen der Nutzer erlernt werden und anhand derer beliebig viele neue Materialmodelle vorgeschlagen werden welche mit der artistischen Vision des Nutzers in Einklang sind. Wir haben dieses System außerdem mit einem neuronalen Renderer erweitert der die akkurate Lichtsimulation um Größenordnungen schneller berechnen kann als herkömmliche Renderer die üblicherweise für diese Art von Aufgabe verwendet werden. Infolge dessen sind unerfahrene sowie erfahrene Benutzer nun in der Lage sehr schnell eine massive Anzahl an Materialien zu synthetisieren, welches für beide eine signifikante Beschleunigung im Lern- und Arbeitsprozess darstellt.

Zweitens, die Simulation von Lichtstreuung unterhalb der Oberfläche eines Materials (subsurface scattering). Solche eine Simulation ermöglicht die realitätsnahe Darstellung bestimmter lichtdurchlässiger Materialien, jedoch haben die meisten publizierten Methoden einen sehr hohen Rechenaufwand und benötigen mehrere Stunden zur Berechnung eines Bildes oder benutzen sehr vereinfachte Annahmen über die Streuung des Lichtes innerhalb des Materials. Wir schlagen eine Reihe von Echtzeitmethoden vor die dieses Problem lösen indem bekannte 2D Faltungsfilter in einzelne 1D Filter zerlegt werden wobei ein hohen Grad an visueller Genauigkeit erhalten bleibt. Diese Methoden haben Laufzeiten von wenigen Millisekunden und können in gängige Renderingsysteme als einfacher Nachbearbeitungsschritt ohne tiefgreifende Änderungen integriert werden.

Abstract

Light transport simulations are the industry-standard way of creating convincing photorealistic imagery and are widely used in creating animation movies, computer animations, medical and architectural visualizations among many other notable applications. These techniques simulate how millions of rays of light interact with a virtual scene, where the realism of the final output depends greatly on the quality of the used materials and the geometry of the objects within this scene. In this thesis, we endeavor to address two key issues pertaining to photorealistic material synthesis: first, creating convincing photorealistic materials requires years of expertise in this field and requires a non-trivial amount of trial and error from the side of the artist. We propose two learning-based methods that enables novice users to easily and quickly synthesize photorealistic materials by learning their preferences and recommending arbitrarily many new material models that are in line with their artistic vision. We also augmented these systems with a neural renderer that performs accurate light-transport simulation for these materials orders of magnitude quicker than the photorealistic rendering engines commonly used for these tasks. As a result, novice users are now able to perform mass-scale material synthesis, and even expert users experience a significant improvement in modeling times when many material models are sought.

Second, simulating subsurface light transport leads to convincing translucent material visualizations, however, most published techniques either take several hours to compute an image, or make simplifying assumptions regarding the underlying physical laws of volumetric scattering. We propose a set of real-time methods to remedy this issue by decomposing well-known 2D convolution filters into a set of separable 1D convolutions while retaining a high degree of visual accuracy. These methods execute within a few milliseconds and can be inserted into state-of-the-art rendering systems as a simple post-processing step without introducing intrusive changes into the rendering pipeline.

Contents

CHAPTER 1

Introduction

1.1 Motivation

Light-transport simulation programs enable artists to synthesize accurate images of virtual scenes by simulating how rays of light interact with the geometry and the materials of these scenes. This process is often referred to as "rendering" an image. With the ascendancy of these techniques, artists became able to create convincing architectural visualizations and computer animations to the point where entire feature-length movies can be rendered solely with virtual actors and objects. However, many of the earlier methods represent objects as *surfaces* – as a result, translucent materials that scatter light within the volume of the objects (e.g., human skin, most plastics, milk, marble and more) remain out of reach. Later, volumetric light-transport techniques opened up the possibility of rendering a variety of these translucent materials, however, they also require significant tradeoffs either in terms of visual quality or have execution times in the order of several hours of days. In this thesis, we identify two key issues that appear in most modern rendering systems and propose a set of solutions to address them:

- Most modern rendering systems offer a node-based shader tool where the artist may adjust a set of physical parameters and obtain a rich selection of photorealistic materials. These are often referred to as "principled" shaders. Their main goal is typically to be able to encapsulate every possible commonly used material model the user might need in a common workflow. However, this expressivity comes with the burden of complexity: the artist has to be able to understand not only the key physical parameters in isolation (e.g., absorption coefficients, indices of refraction), but also how they interact with each other. This requires significant expertise in the field of material modeling.

One of our key observations is that this first issue can be remedied with a learning-based system that enables novice users to perform mass-scale material synthesis without requiring extensive knowledge of physically based light transport. We further speed up this process with a neural renderer that replaces the light transport simulation with a learning technique that can produce faithful images of the recommended materials several orders of magnitude faster. We also propose an intuitive variant-generation method to fine-tune the details of a select set of obtained materials. Finally, we introduce a method that is aimed at novice users that only requires knowledge of the most basic image processing operations. In this proposed workflow, the user starts by applying a few intuitive transforms (e.g., image colorization, image inpainting and more) to a chosen input material, and in the next step, our learning-based technique produces a material model that is faithful to this target image. This process offers a high-quality output material within 20-30 seconds of computation time.

The second issue we're concerned with is the complexity of simulating subsurface light transport:

- Rendering a wide range of translucent materials in real time has been a long-standing challenge in computer graphics. Most existing methods either render only a coarse approximation of subsurface light transport in real time, or take several hours to synthesize an accurate image. The fact that these images have to be re-rendered every time a physical parameter is changed also introduces a non-trivial amount of trial and error in this process and results in an inefficient material modeling workflow.

Convolution-based subsurface scattering methods can be implemented as a post-process filter, providing a promising research direction for this issue. However, most of these methods still remain out of reach for real-time applications where it is a requirement to execute within a few milliseconds. In this thesis, we propose a technique that builds on the observation that many of these 2D convolutions can be decomposed into a set of separable 1D convolutions and be executed significantly faster while retaining a high degree of visual accuracy. This method works in screen space and therefore performs well in real-time environments that are ample in fine geometry and high-resolution textured data.

1.2 Contributions and Publications

The following section breaks down how we apportioned our contributions into three published papers that comprise this thesis and the roles I took in these projects.

- In our **Gaussian Material Synthesis** paper (Chapter 3), we endeavor to address the first issue by proposing a rapid mass-scale material synthesis system for novice and expert users alike. This method learns the preferences of the user by presenting them with a gallery of materials, collecting their scores and recommending arbitrarily many new materials from the learned distributions. However, as each of these recommended materials would take 40-60 seconds to visualize via photorealistic rendering, we also propose a neural renderer that is able to perform these light-transport simulations within a few milliseconds. Neural rendering, in general, remains an unsolved problem. However, one of our key observations is that these material-modeling workflows can take place with a fixed piece of geometry, lighting and camera setup, leading to a restricted problem definition that can be solved efficiently. Finally, in this work, we also propose a latent-space method that enables the rapid exploration of alternatives to a reference material to further reduce the time taken in the laborious fine-tuning stage of the material-modeling process.

In this project, I came up with the mathematical theory, implementation and wrote the entirety of the paper and the supplementary materials. Michael Wimmer and Peter Wonka provided supervision for this work.

> *Károly Zsolnai-Fehér, Peter Wonka, and Michael Wimmer. Gaussian Material Synthesis. ACM Transactions on Graphics (Proc. SIGGRAPH) (2018) 37 (4).*

- Our other work in this direction, **Photorealistic Material Editing Through Direct Image Manipulation** (Chapter 4), offers an improved variant of this technique where artists can reuse their general image-processing expertise to create photorealistic materials more efficiently: first, the user is asked to apply a few intuitive transforms to a reference material (e.g., colorization, image inpainting). In the next step, our technique takes this non-physical image and proposes a photorealistic material that closely approximates it. This method works in the presence of poorly edited images and executes within 20-30 seconds. As a result, the user can skip the exploration step and arrive at their envisioned material model immediately, within one step.

  In this project, I came up with the mathematical theory, implementation and wrote the entirety of the paper and the supplementary materials. Michael Wimmer and Peter Wonka provided supervision for this work.

  > *Károly Zsolnai-Fehér, Peter Wonka and Michael Wimmer. Photorealistic Material Editing Through Direct Image Manipulation (Technical Report, under review).*

- With **Separable Subsurface Scattering** (Chapter 5), we endeavor to address the problem of efficient subsurface light transport by proposing a set of real-time techniques, all of which rely on separable approximations. In this work, we show that this effect can be simulated as a collection of rapid 1D separable convolutions in real time while delivering results of higher quality than what previously known techniques offer. To maximize impact, we designed these techniques in a way that they can be implemented as simple post-processing steps without introducing intrusive changes to the rendering pipeline.

  I have worked on the mathematical theory, wrote the majority of the paper, and supervised the implementation of this project. The initial idea is the work of

Jorge Jimenez, who produced the first implementation and a tech report with Adrian Jarabo, which was extended later from our side by Christian Freude. The pre-integrated model was derived by Thomas Auzinger, and the project was jointly supervised by Diego Gutierrez and Michael Wimmer.

Jorge Jimenez, Károly Zsolnai, Adrian Jarabo, Christian Freude, Thomas Auzinger, Xian-Chun Wu, Javier von der Pahlen, Michael Wimmer, and Diego Gutierrez. Separable subsurface scattering. Computer Graphics Forum (2015) 34 (6).

All three of the showcased papers are equivalent to their published version, with a few changes to provide a better flow for the thesis, i.e., the appendices, parts of the supplementary material and Q&A-s have been placed into their corresponding papers. To foster future works in this area, we provide the full source code and pre-trained neural networks for the entirety of all of these projects.

1.3 Technical Background and Challenges

In this thesis, our main goal is to provide a set of tools to empower artists to be able to create their envisioned photorealistic materials without requiring years of expertise in material modeling. This chiefly refers to learning user preferences and simplifying the material-modeling pipeline, noting that a prerequisite of this process is also that artists can rapidly produce new images of their chosen materials (this process is commonly referred to as *rendering*). In this section, we provide a gentle introduction to the basics of light transport and pinpoint how our contributions fit in this research field.

1.3.1 The BRDF

In order to synthesize a photorealistic image, a simulation framework must contain a mathematical description of the scattering properties of materials that appear in nature, which typically takes places through the Bidirectional Reflectance Distribution Function (BRDF), i.e.,

$$f_r\left(\vec{\omega}, x, \vec{\omega}'\right), \tag{1.1}$$

which is a probability density function that outputs the probability of an outward direction $\vec{\omega}'$ when given an incoming light direction $\vec{\omega}$ at a spatial position x. This provides a simple and elegant way of describing the more rudimentary materials seen in nature, e.g., a diffuse (matte) material can be represented with a uniform distribution, and a perfect specular mirror can be thought of as the Kronecker delta function which is 1 at the perfect reflection direction and 0 everywhere else. This BRDF term is to be evaluated every time a ray of light intersects a material within a scene and admits three key properties that makes working with it easier. Below, we formally describe these properties and provide an intuitive explanation for each of them:

- The BRDF obeys the **Helmholtz-reciprocity** rule, i.e.,

$$\forall \vec{\omega}, \vec{\omega}' : \quad f_r\left(\vec{\omega}, x, \vec{\omega}'\right) = f_r\left(\vec{\omega}', x, \vec{\omega}\right). \tag{1.2}$$

  Intuitively, this means that in our formulations, the direction of a ray of light may be reversed and all of our previous mathematical calculations will remain correct. Normally, we build light paths from from light source towards the camera – this is a solution that is computationally wasteful due to the fact that most rays of light in the scene never end up hitting the camera. Instead, we can start them from the camera, guaranteeing that at least one of the endpoints of every ray of light contributes to our final image. Helmholtz-reciprocity is useful in these cases, and tracing rays this way is thus the baseline for the most commonly used global illumination methods.

- **Positivity**,

$$\forall \vec{\omega}, \vec{\omega}' : \quad f_r\left(\vec{\omega}, x, \vec{\omega}'\right) \geq 0. \tag{1.3}$$

  This property ensures that it is impossible for an incoming-outgoing direction pair to have a negative probability.

- **Energy conservation**. A material may reflect and absorb a portion of the incoming light, but the sum of the outgoing energy may not be greater than the total amount of incoming energy. Formally,

$$\int_{\Omega} f_r\left(\vec{\omega}, x, \vec{\omega}'\right) \cos\theta \; d\vec{\omega}' \leq 1, \tag{1.4}$$

  where the presence of the cosine term accounts for the attenuation of light rays arising from the angle (θ) between the incoming direction and the surface normal. The integration takes places on Ω, i.e., the illumination hemisphere. In the case of a perfectly reflective material, the left side equals 1, and for perfectly absorbing blackbodies, it will equal 0. In a simulation that violates this property, the radiance transferred by a ray of light can increase upon every bounce, even in perpetuity (depending on the maximum number of simulated bounces), often resulting in a perfectly white output image.

So far, we have discussed rays of light bouncing off of (or absorbed by) different materials and how the BRDF describes this behavior. We note that different variants of this probability density function also exist, e.g., the Bidirectional Transmittance Distribution Function (BTDF) to account for transmissive materials, and that the term BSDF (S is for Scattering) is used as a common way of referring to a family of these functions, i.e., $BSDF = BRDF \cup BTDF$.

1.3.2 Difficulties and Solutions

From the viewpoint of a physicist, BSDF models provide an excellent way of modeling physically based real-world materials. In fact, the concept itself is so powerful that

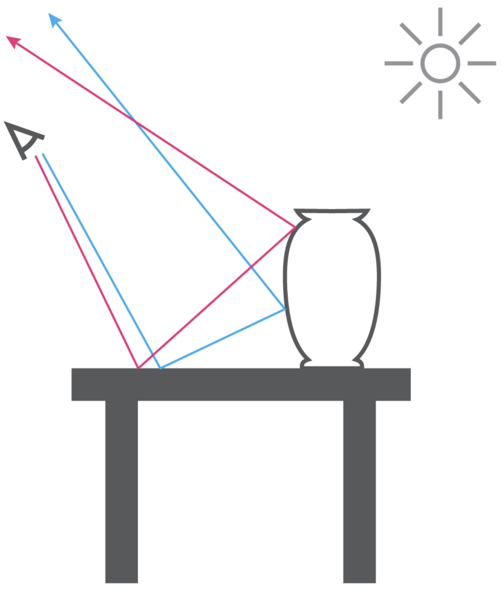


Figure 1.1: A standard simulation of the rendering equation assumes no participating medium and bounces rays of light off of the surface of solid objects.

a “principled” BSDF model [BS12] can be built that is typically implemented as an interpolation of arbitrarily many already existing BSDF models (e.g., diffuse, specular, translucent) and hence, can span many common materials appearing in our daily lives. However, such a principled model depends on a vast number of physical parameters, and as many of these photorealistic rendering programs end up in the hands of artists who do not have the necessary background knowledge of the intricacies of light-transport may struggle in productively using these principled material models. To alleviate this, in **Gaussian Material Synthesis**, we set out to create a tool that presents the artist with a collection of example materials and asks for a set of scores to be able to learn which ones are close to their envisioned results, and recommend new materials that the artist is expected to find desirable. Alternatively, in our work, titled **Photorealistic Material Editing Through Direct Image Manipulation**, we offer a different learning-based solution that skips this step and immediately shows the closest achievable BSDF for an edited input image.

1.3.3 The Rendering Equation

The BRDF term, as we described in the previous section, cannot yield output images unless it is embedded within a rendering system. To be able to create an image with a high degree of realism, many millions of light rays have to be simulated and bounced around in the scene (Fig. 1.1). In the case of path tracing (the unidirectional kind), these rays originate at the camera and traverse the scene in a straight line until they intersect a solid object. In this case, the main quantity of interest is $L_o(x, \vec{\omega})$, i.e., the exiting radiance (measured in $\mathrm{W} \cdot \mathrm{sr}^{-1} \cdot \mathrm{m}^{-2}$) from a spatial position x towards $\vec{\omega}$, i.e., an incoming direction (typically pointing towards the camera). The rendering equation records that this quantity is obtained as the sum of the emitted light towards the camera $L_e(x, \vec{\omega})$ and the reflected incoming light $L_i\left(x, \vec{\omega}'\right)$ weighted by the BRDF and a cosine term for light attenuation [Kaj86], i.e.,

$$L_o(x, \vec{\omega}) = L_e(x, \vec{\omega}) + \int_{\Omega} L_i\left(x, \vec{\omega}'\right) f_r\left(\vec{\omega}, x, \vec{\omega}'\right) \cos\theta \, d\vec{\omega}', \tag{1.5}$$

where θ denotes the angle between the incoming light direction and the surface normal, $\int_{\Omega} \ldots d\vec{\omega}'$ denotes an integral operator for the illumination hemisphere over all possible $\vec{\omega}'$ incoming light directions. Solving this Fredholm integral equation of the second kind is fraught with difficulties: a closed-form solution can only be obtained in a small set specialized cases, partly due to the mathematical complexity of the integrand (e.g., scene geometry, visibility, discontinuities), and beyond that, the inner $L_i\left(x, \vec{\omega}'\right)$ also expands into a separate rendering equation. As the potential number of light-ray bounces is unbounded, this process yields in an infinite-dimensional integral. The infinite-dimensional nature of the integral can be addressed by using Russian Roulette, a technique that terminates light paths without introducing systemic bias [PH04], or alternatively, a simpler, but biased solution can be obtained by imposing a limit on the number of light ray bounces in the simulation. Furthermore, if the BRDF term describes a perfectly specular material (i.e., a Kronecker delta), the integrand becomes singular. This can be partly addressed with an "if this then that" solution in which the output ray direction is selected deterministically as the perfect reflection direction. More sophisticated solutions can also be obtained through regularization [KD13], i.e., creating a mollification function that "smears" the infinitely thin spike of a perfect specular reflection into a more well-behaved diffuse BRDF that is easier to handle early on in the rendering process. In this case, the amount of this "smearing" is referred to as *mollification width*, which asymptotically tends to zero as the simulation progresses, thereby providing an unbiased simulation as the number of drawn samples tends to infinity.

1.3.4 Difficulties and Solutions

As noted, the rendering equation, in general, cannot be solved analytically, hence, the most common solutions are obtained through Monte Carlo integration. This technique takes the average of samples drawn randomly from the integral, and over time, a "good enough" solution emerges, provided that a sufficient amount of samples are used. Initially, as the first few samples are taken, depending on the random choices, the estimation of the integral is under or overestimated, and these inaccuracies materialize in the output image as noise. As more samples are added, the error of a consistent Monte Carlo estimator asymptotically converges to zero, i.e., sooner or later, a perfectly noise-free output image is produced. However, as the expected error of the Monte Carlo estimation using N samples is proportional to $1/\sqrt{N}$, 4 times as many samples are required to cut the error down to 50%, 16 times as many are required to cut down the error to 25%, and 100 times as many

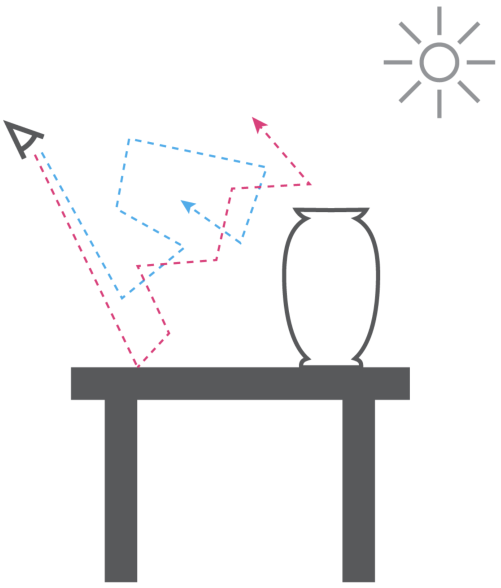


Figure 1.2: The radiative transport equation introduces participating media in the simulation process, and thus, intersection points can appear anywhere in space and need not to be attached to object boundaries.

samples are needed to obtain a result that is 10 times better. With the ascendancy of neural network-based learning methods, a few early works had shown that a few aspects of light transport, e.g., indirect illumination, can be learned and resolved via neural networks with remarkable accuracy [RWG+13]. Later, neural network-based solutions were used to take over a greater set of light-transport phenomena [NAM+17b], however, general neural rendering is still not a possibility. In our **Gaussian Material Synthesis** and **Photorealistic Material Editing Through Direct Image Manipulation** works, we recognized that in most material-editing workflows, the scene geometry, lighting and camera positions can be kept constant, in which case, the light-transport simulation can be substituted by a neural renderer that is able to produce images that are close to the simulated ground-truth solutions. However, instead of taking from minutes to hours to compute, the neural network infers a new image orders of magnitude quicker, i.e., within 2-6 milliseconds. This way, we were able to create a material-modeling workflow where the artist can immediately observe the results of our material-synthesis methods, which, according to our studies, substantially enhances their productivity.

1.3.5 The Radiative Transport Equation and BSSRDF

Unless special measures are taken (e.g., BSSRDFs are used instead of BSDFs, to be discussed later in this section), the rendering equation assumes no participating medium, and hence, all intersection points are located on the surfaces of solid objects. This simplification bars it from simulating a number of volumetric phenomena, such as haze, fog, smoke, and a rich selection of translucent materials, such as human skin, milk, marble slabs, and more. The radiative transport equation takes these effects into account (Fig. 1.2), i.e.,

$$(\vec{\omega} \cdot \vec{\nabla})L(x,\vec{\omega}) = -\,\sigma_t L(x,\vec{\omega}) + Q(x,\vec{\omega}) + \sigma_s \int_{4\pi} L(x,\vec{\omega}) p\left(\vec{\omega},\vec{\omega}'\right) \mathrm{d}\vec{\omega}' \quad . \tag{1.6}$$

Note that this formulation yields not the radiance $L(x,\vec{\omega})$ itself, but its change along the direction $\vec{\omega}$, which decreases when traveling through an absorbing (σ_a) and scattering (σ_s) medium (where the extinction coefficient is given as $\sigma_t = \sigma_a + \sigma_s$), and potentially increases through a source term $Q(x,\vec{\omega})$, and via in-scattering from all possible directions, i.e., $\int_{4\pi} \ldots \mathrm{d}\vec{\omega}'$. The phase function $p\left(\vec{\omega},\vec{\omega}'\right)$ can intuitively be thought of as the participating-media equivalent of the BRDF and describes the scattering behavior of the medium. Here, we assume it to be a normalized, i.e.,

$$\int_{4\pi} p\left(\vec{\omega},\vec{\omega}'\right) d\omega' = 1, \tag{1.7}$$

and only dependent on the phase angle,

$$p\left(\vec{\omega},\vec{\omega}'\right) = p\left(\vec{\omega} \cdot \vec{\omega}'\right), \tag{1.8}$$

and thus independent of spatial position x. In the case of an infinitesimally thin beam traveling a distance s in direction $\vec{\omega}_i$ within a homogeneous medium, the radiative transfer equation simplifies to an exponentially decaying function,

$$L_s\left(x_i + s\vec{\omega}_i, \vec{\omega}_i\right) = e^{-\sigma_t s} L_i\left(x_i, \vec{\omega}_i\right). \tag{1.9}$$

This is an intuitive property of volumetric light transport as a flashlight can easily be seen through a sheet of paper, remains faintly visible when hidden behind a human ear or finger, but gets completely concealed behind a thick slab of marble. This formulation accounts for a wide swath of participating media and translucent materials, however, further increases the dimensionality of the integral *(per bounce)* and hence, the number of Monte Carlo samples required to solve for L often becomes intractable, especially if we take into account multiple scattering.

In many practical applications, the scene itself doesn't require an external participating medium such as haze or smoke, but contains translucent materials. In these cases, one can obtain most of the advantages of the radiative transport equation by reverting to the rendering equation and switching the BRDF terms to the bidirectional surface scattering distribution function (BSSRDF) [JMLH01],

$$L_o\left(x_o, \vec{\omega}_o\right) = \int_A \int_{2\pi} S\left(x_i, \vec{\omega}_i; x_o, \vec{\omega}_o\right) L_i\left(x_i, \vec{\omega}_i\right) \cos\theta \, d\omega_i dA\left(x_i\right). \tag{1.10}$$

Note that in this case, it is possible to eschew using the radiative transfer equation and solve a local surface integral instead. The BSSRDF is a 4-dimensional function where

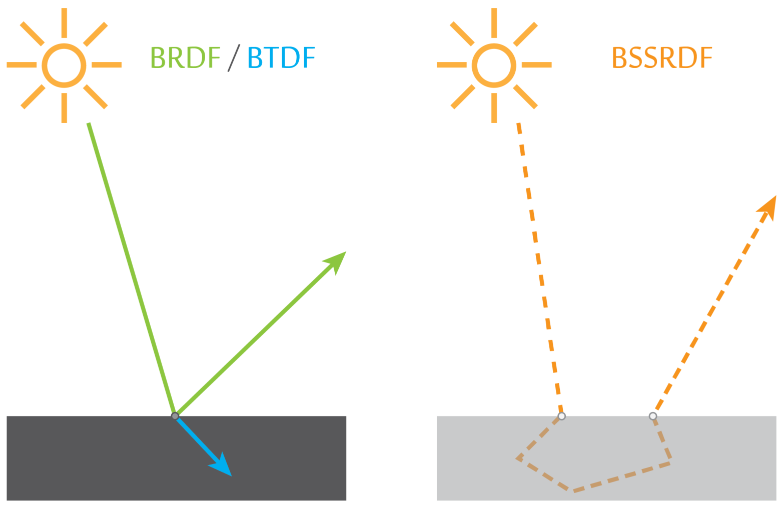


Figure 1.3: A simulation that includes computing subsurface light scattering and absorption events through the BSSRDF (right) opens up the possibility of visualizing a rich selection of translucent materials.

a ray of light is assumed to enter the volume at point x_i and direction $\vec{\omega}_i$, undergoes a potentially vast number of scattering events (or gets absorbed), and exits at point x_o and direction $\vec{\omega}_o$ (Fig. 1.3). However, this all comes at a price: to obtain the outgoing radiance $L_o\left(x_o, \vec{\omega}_o\right)$, this expression is subject to integration over incoming directions $\int_{2\pi} \ldots d\omega_i$ and the surface area subject to subsurface light transport, i.e., $\int_A \ldots dA\left(x_i\right)$.

1.3.6 Difficulties and Solutions

The BSSRDF formulation enables us to keep using the rendering equation and still simulate translucent materials, but due to its increased integration requirements, it is still computationally expensive, i.e., often takes from minutes to hours to converge for practical cases. To alleviate this, a few reasonable simplifications can be made to accommodate many creative applications that require the visualization of these translucent materials. First, if one lights a normally incident, infinitesimally thin pencil beam towards an infinite half-space made of a homogeneous translucent material, a symmetric diffusion profile (R_d) emerges (Fig. 1.4). This profile is useful when formulating the diffusion term (S_d) of the BSSRDF [JMLH01], i.e.,

$$S_d\left(x_i, \vec{\omega}_i; x_o, \vec{\omega}_o\right) = \frac{1}{\pi} F_t\left(\eta, \vec{\omega}_i\right) R_d\left(\left\|x_i - x_o\right\|\right) F_t\left(\eta, \vec{\omega}_o\right), \tag{1.11}$$

where the Fresnel transmittance term (F_t) describes the probability of transmittance upon incidence between two different optical media (given by their relative indices of refraction η) and the diffusion profile only depends on the distance r, i.e., $R_d(r) = R_d\left(\left\|x_i - x_o\right\|\right)$. Intuitively, the first term accounts for the pencil beam entering the volume, the second

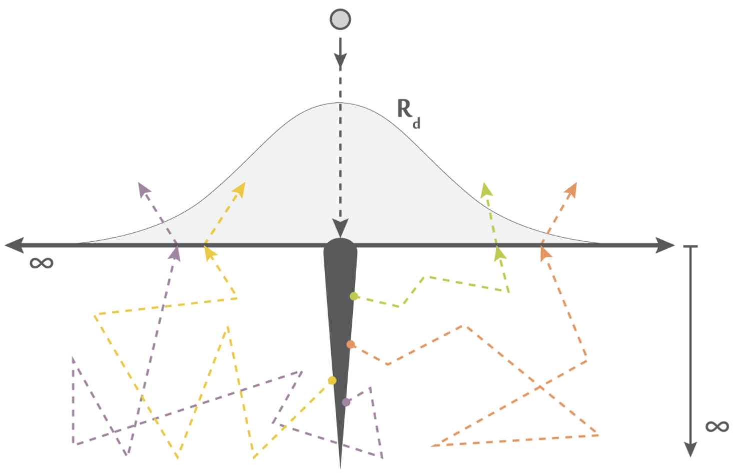


Figure 1.4: The diffusion profile emerging over a homogeneous material from a normally incident, infinitesimally thin pencil beam. The design of this image was inspired by Habel et al.'s work [HCJ13].

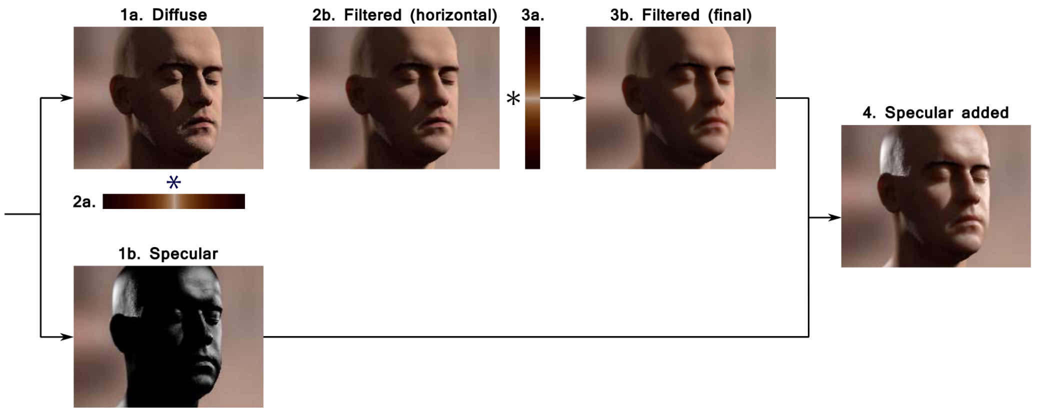


Figure 1.5: Our work adds subsurface light transport to the irradiance signal as a simple post-processing step. Source: Christian Freude's corresponding thesis on our project [Fre15].

term expresses the amount of scattering between the entrance and exit points, and the third term accounts for the beam exiting the volume.

These diffusion profiles can be recorded for a variety of scattering materials through a Monte Carlo simulation [WJZ95] once and be reused in perpetuity. We showcase the result of such a simulation for a skim milk material in Fig. 1.6. The execution time of the rendering process can be further improved by approximating R_d via a sum of Gaussians and taking advantage of the fact that the 2D radial convolution of two Gaussians will

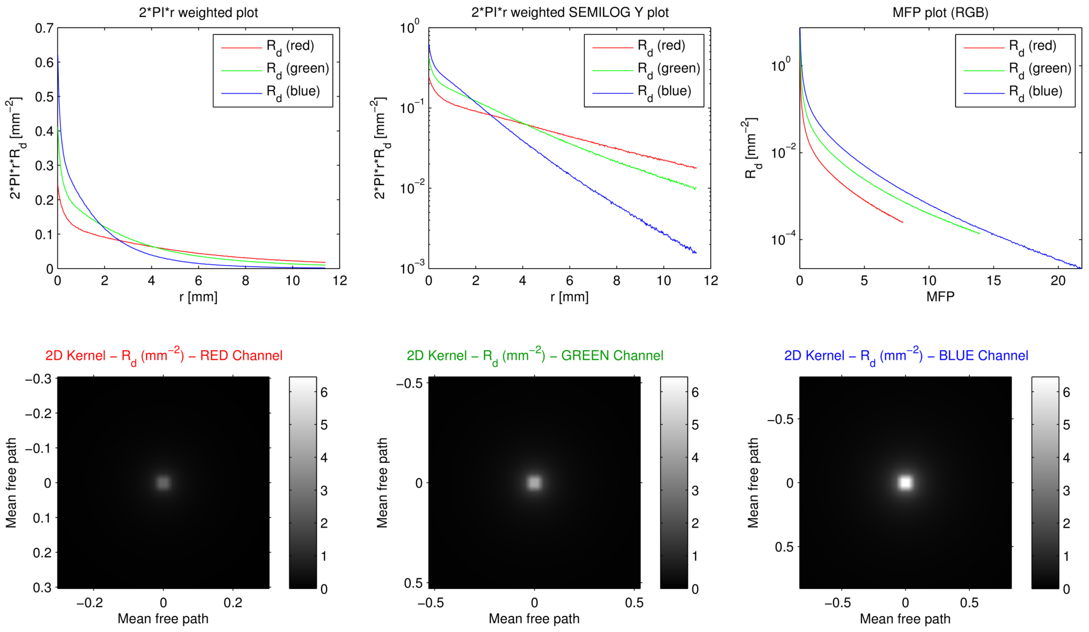


Figure 1.6: Our simulation of the diffusion profile of skim milk on the three representative wavelengths. Note that as $R_d(r)$ is symmetric for isotropic, homogeneous materials, it only depends on the distance r and it is sufficient to visualize one half of the distribution.

always yield another Gaussian [dLE07]. Such a 2D convolution kernel can be applied to the underlying irradiance signal and will process orders of magnitude faster than a comprehensive volumetric simulation. With **Separable Subsurface Scattering**, we endeavored to contribute to this line of research by proposing a set of fast, separable 1D convolutions to bring the computational requirement of this process further down to 0.489ms per frame for a full HD image while retaining the visual quality of the solution. This technique also works in screen space, can be applied as a simple post-processing step (Fig. 1.5), does not require intrusive architectural changes to the renderer and is actively used in the industry for several AAA titles created by Activision-Blizzard.

CHAPTER 2

Related Work

In our first two works, we are primarily concerned with improving the process of material modeling for artists working with photorealistic rendering tools, while our third technique improves the rendering of real-time subsurface scattering. In the following, we provide a brief overview of the most pertinent related research works.

2.1 Material Acquisition

Many of these workflows start with an acquisition step where a real-world material is to be measured with strobes and turntables [MIWI16], screens and cameras [AWL13] or other equipment to obtain a digital version of it that mimics 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$^+$13, RJGW19] or approximated through an analytic BSDF model [PdMJ14]. Many learning-based works aim to improve the cost efficiency and convenience of this acquisition step by only requiring photographs of the target material [AWL$^+$15, AAL16, DAD$^+$18, DAD$^+$19, GLD$^+$19]. Alternatively, precomputed BSDF databases offer the possibility of working with these materials without the acquisition of a physical copy [Mat03, DJ18], where the key challenge is easing the navigation and browsing of the high-dimensional measured data. Database-driven methods contain this acquired reflectance data for a vast number of possible materials to populate a scene, however, they are typically very sizable and either cannot produce new materials on the fly [BUSB13], or are lacking in more sophisticated material representations (e.g., BSSRDFs) [Mat03]. These high-dimensional datasets can be transformed onto a lower-dimensional manifold, where the navigability of the measured data can be improved through interpolation [SSN18]. We discuss further techniques for dimensionality reduction in Section 2.5 and demonstrate practical use-cases in our Gaussian Material Synthesis work.

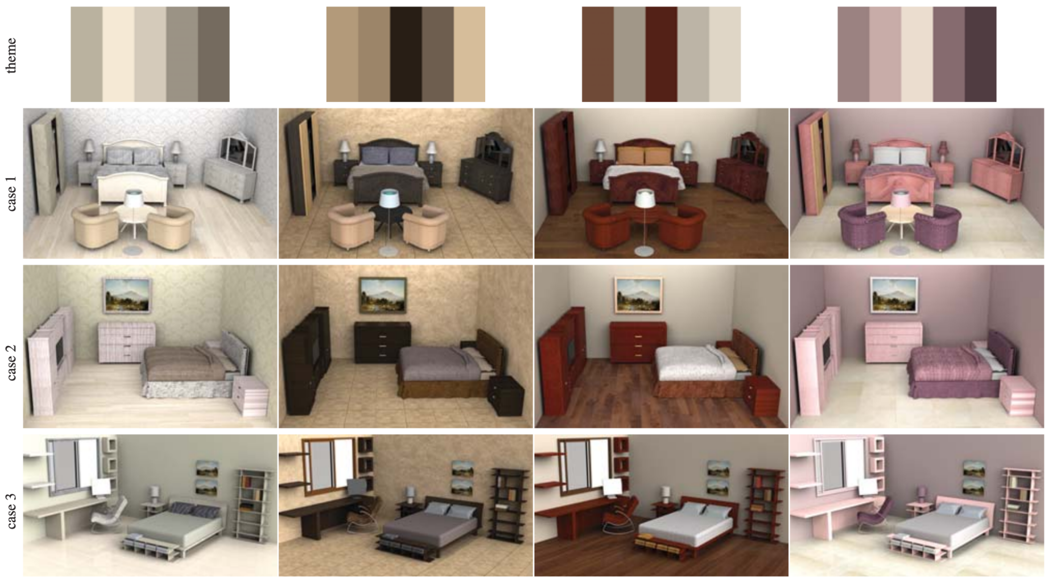


Figure 2.1: Chen et al.'s work [CXY+15] (also the source of this image) recommends a set of material models for indoor scenes that satisfy a user-specified color scheme.

In our first two works, we aimed to develop two novel systems where no physical access to the sought materials or additional equipment is required, and where fine artistic control over the outputs is a possibility. In addition to the BRDF recovery methods mentioned earlier, our proposed material synthesis techniques are also related to inverse rendering approaches [MG98, RH01], where important physical properties are inferred from a real photograph, typically with unknown lighting conditions. In our works, the material test scene contains a known lighting and geometry setup, enabling not only the rapid discovery of new materials, but artistic control through standard and well-known image-space editing operations.

2.2 Material Editing

When a physical copy of a sought material is not available, the artist may, instead, use an already existing material database or an appropriately expressive principled shader, both of which are available within many photorealistic rendering systems. To be able to efficiently use these systems, an artist is typically required to have an understanding of physical quantities pertaining to the most common modeled phenomena in light transport, e.g., indices of refraction, scattering and absorption albedos and more [STPP09, BS12]. This modeling time can be cut down by techniques that enable editing 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. To alleviate this, our first framework does not expose any physical BSDF parameters to the user, but learns

Name	In-scene editing	Exploration	Moderate-scale	Mass-scale
Direct interaction	✗	✗	✗	✗
BRDF relighting	✓	✗	✗	✗
Gaussian Material Synthesis (Chapter 3)	✗	✓	✓	✓
Photorealistic Material Editing (Chapter 4)	✗	✓	✓	✓

Table 2.1: The key advantage of BRDF relighting methods is the possibility of editing the materials directly within the final scene at the cost of forfeiting rapid exploration and mass-scale material synthesis. To help the artist explore many potential candidate materials, GMS supports variant generation through a 2D latent-space projection while our Photorealistic Material Editing (PME) method offers close to real-time performance on image sequences. The yellow and green check marks showcase that PME outperforms in moderate-scale problems while GMS excels at mass-scale material synthesis.

the user preferences directly and is able to rapidly recommend desirable material models on a mass scale. Reducing the expertise required for material editing workflows has been a subject to a significant volume of research works: an intuitive editor was proposed by pre-computing many solutions to enable rapid exploration [HR13], and carefully crafted material spaces were derived to aid the artist [SGM^{+}16, SSN18, LMS^{+}19]. As an alternative solution, selectively editing physical properties such as albedos, specular reflections without changing other parameters is also a possibility [SJR18, TKL^{+}16]. The separation of these parameters during editing is highly beneficial, especially for diffuse-specular components [SJR18]. Instead of editing these materials directly, other techniques enable editing secondary effects, such as caustics and indirect illumination within the output image [SNM^{+}13].

With our Photorealitic Material Editing work, we 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. Chen et al.'s work [CXY^{+}15] (Fig. 2.1) operates by populating a scene with materials based on an input color scheme. We expect that this, along with other color mixing methods (e.g., Shugrina et al.'s work [SLD17]) can be combined with our proposed techniques to enhance the process of assigning a collection of materials to a scene. By using our Photorealistic Material Editing system, artists can reuse their image editing knowledge and apply it to material synthesis, even if they don't have any direct experience in this field. If one, or at most a handful of materials are sought, the modeling times of our proposed method are preferable to Gaussian Material Synthesis (GMS). In Table 2.1, we endeavored to simplify the process of choosing the appropriate class of methods for a prescribed application.

2.3 Neural Networks and Rendering

The recent resurgence of neural network-based learning techniques stimulated a large body of research works in photorealistic rendering. A class of techniques uses neural networks to approximate a selected aspect of light transport such as indirect illumination [RWG+13] or participating media [KMM+17]. To extend these endeavors, other works can be used to perform other related tasks, such as approximating sky models [SBRCD17] or Monte Carlo noise filtering [GLA+19, LMH+18, KBS15]. There is also a growing interest in replacing a greater feature set of the renderer with learning algorithms [NAM+17a], which typically requires the presence of additional information, e.g., a number of auxiliary buffers. In our problem formulations, we are interested in a restricted version of this problem where geometry, lighting, and the camera setup are fixed and the BSDF parameters are subject to change. We show that in this case, it is possible to replace the entirety of the renderer without a noticeable loss of visual quality.

2.4 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]. Later, in our photorealistic material editing 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. 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 [ZKSE16]. Zhu et al.'s method uses a generative model to synthesize images, whereas our PME technique seeks a parameter setup to be used with a principled shader. In their work, the space of image-editing operations is constrained, but in return, yields a large variability for their output images. Our technique strikes a different tradeoff where the space of editing operations is more forgiving, and produces outputs that must adhere to the rules of the principled shader, i.e., represent photorealistic materials. This design choice also necessitates our "best of 9" scheme to provide robust results. Furthermore, our optimization process involves invoking a neural renderer to produce the intermediate images to compare against the target image, and each of our stages are modular, i.e., can be used in isolation or combined together depending on the requirements of the artist.

2.5 GPR, GPLVM

Gaussian Process Regression (GPR) is an effective learning method where prior knowledge of a problem can be harnessed via a covariance function, enabling high-quality regression using a modest amount of training samples. It offers useful solutions in the perimeter of computer graphics and machine learning – examples include performing super resolution [HS11], analyzing and generating dynamical models for human motion [WFH08], or synthesizing doodles and ocean waves [AL11]. Generative latent-space techniques proved to be highly useful in a variety of areas: they are able to design new fonts by using the *Gaussian Process Latent Variable Model* (GPLVM) to find low-dimensional structures in high-dimensional data and expose them to the user [CK14], generate imaginary human faces and perform meaningful algebraic operations between them [BJLPS17], synthesize new shapes when given a database of examples [AKZM14], or suggest a selection of perceptually different parameter choices [MAB^{+}97, KSI14]. It is clear that these methods are powerful tools in isolation – in this thesis, we show a novel combination of GPR, GPLVM and a Convolutional Neural Network that opens up the possibility of learning the material preferences of a user and offering a 2D latent space where the real-time fine-tuning of the recommended materials is possible.

2.6 Off-line Subsurface Scattering

The simulation of scattering inside translucent materials dates back to the radiative transfer equation [Cha60], which can be solved by traditional path-sampling techniques. The solution of this integral is a very demanding process in terms of computation time, especially if solved for a high number of bounces. Optimization techniques to reduce the computation times include using a dipole model [JMLH01, JB02] and modeling multiple scattering as a diffusion process [Sta95]. Donner and Jensen [DJ05] extended the dipole into a multipole model that allows modeling multi-layered translucent materials, such as skin. The same authors later introduced a photon diffusion technique to combine photon tracing and the diffusion approximation [DJ07]. These works yield impressive results with computation times in the order of seconds per image. The inherent inaccuracies of the diffusion theory can be overcome by using a more refined diffusion model to separate single and multiple scattering terms, alongside with the quantization of the Green's function of the diffusion equation to obtain realistic all-frequency results [DI11] (Fig. 2.2). Further improvements revolve around better importance sampling [KF12], while a different class of techniques relies on solving the searchlight problem by means of Monte Carlo integration imbued by multiple importance sampling [HCJ13].

2.7 Real-time Subsurface Scattering

Borshukov and Lewis [BL03] approximate subsurface scattering by blurring a 2D diffuse irradiance texture using a Gaussian filter. While this technique is efficient and maps well to the GPU, it neglects the more subtle details of subsurface scattering. This idea

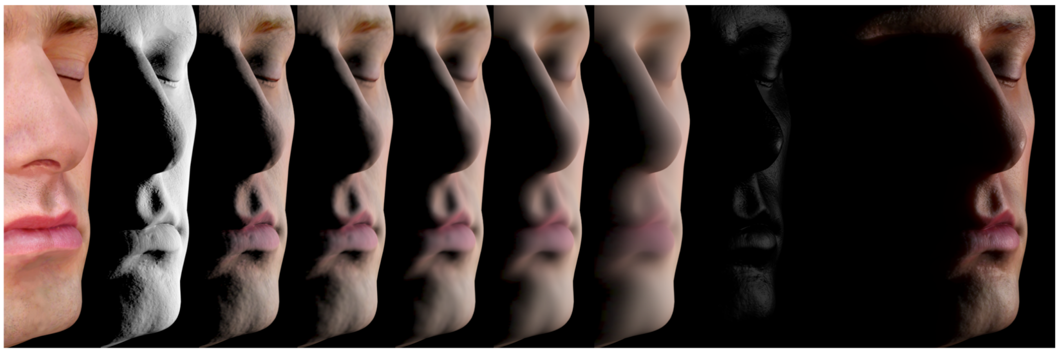


Figure 2.2: d'Eon et al.'s seminal work on subsurface scattering [DI11] (also the source of this image) adds the diffusion effect to the albedo and irradiance signals (1st and 2nd layers) by convolving a set of of Gaussians (3rd to 7th layers) and combines it with specularity information (8th layer) to produce a high-quality final image (9th layer).

is later extended by d'Eon et al. [dLE07, dL07] to develop a high-quality real-time skin shader. They approximate the multipole model with a sum of Gaussians, and use them to blur the irradiance signal in texture space. Since the Gaussians are separable, this allows transforming the expensive 2D convolutions into a cheaper set of 1D convolutions. This technique enables real-time frame rates, while giving results that are comparable to offline simulations. In follow-up work, additional optimizations are introduced, based on computing a single 2D convolution at 13 jittered sample points, which account for direct reflection and two levels of scattering [HBH09]; unfortunately, 13 samples are not enough for a 2D convolution, which leads to poor results.

Although these techniques provide real-time frame rates, they scale poorly with the number of translucent objects in the scene, since the subsurface-scattering simulation needs to be performed on a per-object basis. To overcome this, Jimenez et al. [JSG09, JG10] propose to translate the simulation from texture to screen space. The diffuse reflection of the translucent object is blurred as a post-processing step employing the sum-of-Gaussians formulation, thereby limiting subsurface scattering computations to the visible parts of the objects. Other techniques operating in screen space include the work of Mertens et al. [MKB$^+$05], using importance sampling of the BSSRDF, Shah et al. [SKP09], who use a splatting process for integration instead of a gathering step, and Mikkelsen [Mik10], who shows how convolution with a Gaussian can be expressed as a cross bilateral filter. Penner and Borshukov [PB11] pre-integrate the illumination effects of subsurface scattering due to curvature and shadowing into textures, assuming that surface normals can be pre-blurred, and that soft shadows are used. Recent real-time techniques use the diffusion approximation to render optically thick materials [WZT$^+$08], including using finite elements and finite differences to support arbitrary, non-locally flat geometry [WWH$^+$10, LSR$^+$13]. In contrast to these works, our Separable Subsurface Scattering technique defers the blurring until the shading has been computed, thus retaining all the geometric detail. Furthermore, it is simpler to implement and yields very high frame rates, suitable for the most challenging real-time scenarios.

CHAPTER 3

Gaussian Material Synthesis

3.1 Motivation

Mass-scale photorealistic material synthesis is a labor-intensive endeavor where 3D artists are asked to create a vast number of material variants with similar properties (e.g., metals, minerals, or glassy materials). This requires a fair bit of trial and error and significant domain expertise and is typically left to experts. In this work, we endeavored to address the difficulties raised in Sections 1.3.2 and 1.3.4, i.e., enabling laypersons to be able to operate a "principled" shader without any expertise in light transport, and speeding up the rendering process during material modeling. Thus, we present a learning-based system for rapid mass-scale material synthesis that is useful for novice and expert users alike[1]. The user preferences are learned via Gaussian Process Regression and can be easily sampled for new recommendations. Typically, each recommendation takes 40-60 seconds to render with global illumination, which makes this process impracticable for real-world workflows. Our neural network eliminates this bottleneck by providing high-quality image predictions in real time, after which it is possible to pick the desired materials from a gallery and assign them to a scene in an intuitive manner. Workflow timings against Disney's "principled" shader reveal that our system scales well with the number of sought materials, thus empowering even novice users to generate hundreds of high-quality material models without any expertise in material modeling. Similarly, expert users experience a significant decrease in the total modeling time when populating a scene with materials. Furthermore, our proposed solution also offers controllable recommendations and a novel latent space variant generation step to enable the real-time fine-tuning of materials without requiring any domain expertise.

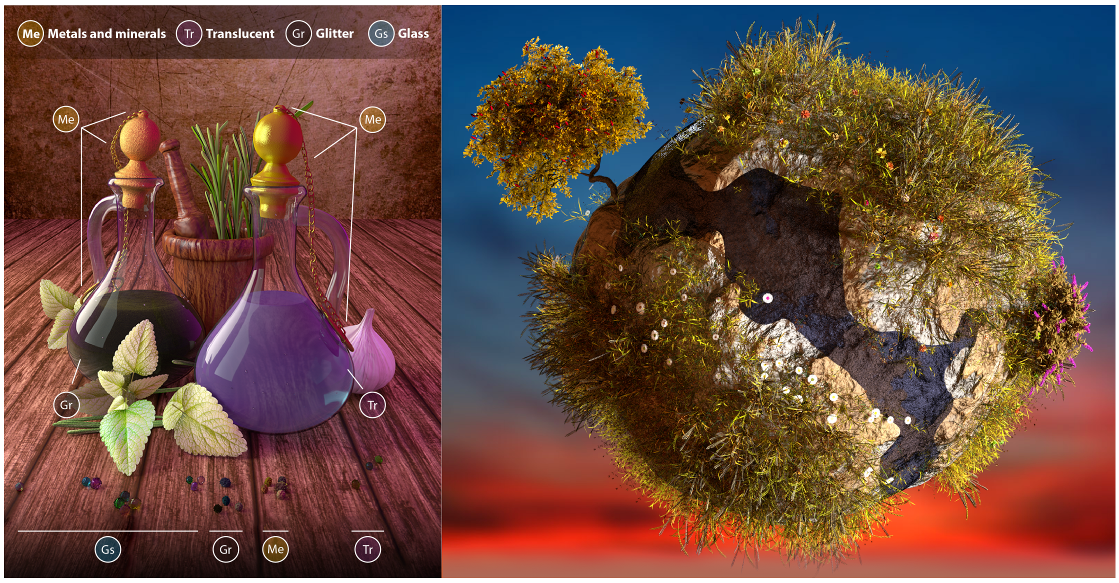


Figure 3.1: Our system opens up the possibility of rapid mass-scale material synthesis for novice and expert users alike. This method takes a set of user preferences as an input and recommends relevant new materials from the learned distributions. On the left, we populated a scene with metals and minerals, translucent, glittery and glassy materials, each of which was learned and synthesized via our proposed technique. The image on the right showcases rich material variations for more than a hundred synthesized materials and objects for the vegetation of the planet. The learning and recommendation steps take less than a minute. The following materials were synthesized for the Microplanet scene: dandelions (upper part of the planet, high color variation), daisies (the white color is fixed, the core follows a slight color variation), staghorn tree (upper left), sweet pepper bush (lower right), Kentucky blue grass and rye (general vegetation covering the planet), the water stream in the middle (one material, extended shader). The following materials were given: the bark of the staghorn tree in the upper left, the procedural dirt material on the surface of the planet and the background HDR image.

3.2 Introduction

Light transport simulations are the industry standard way to create high-quality photorealistic imagery. This class of techniques enjoys a variety of use in architectural visualization, computer animation, and is rapidly becoming the choice of many mass media and entertainment companies to create their feature-length films. Beyond using physically accurate algorithms, the presence of complex material models and high-resolution geometry are also important factors in creating convincing imagery. Choosing the perfect material models is a labor-intensive process where an artist has to resort to trial and error, where each try is followed by the lengthy process of rendering a new image. In this work, we

[1]This chapter is based on our equivalently named paper [ZFWW18].

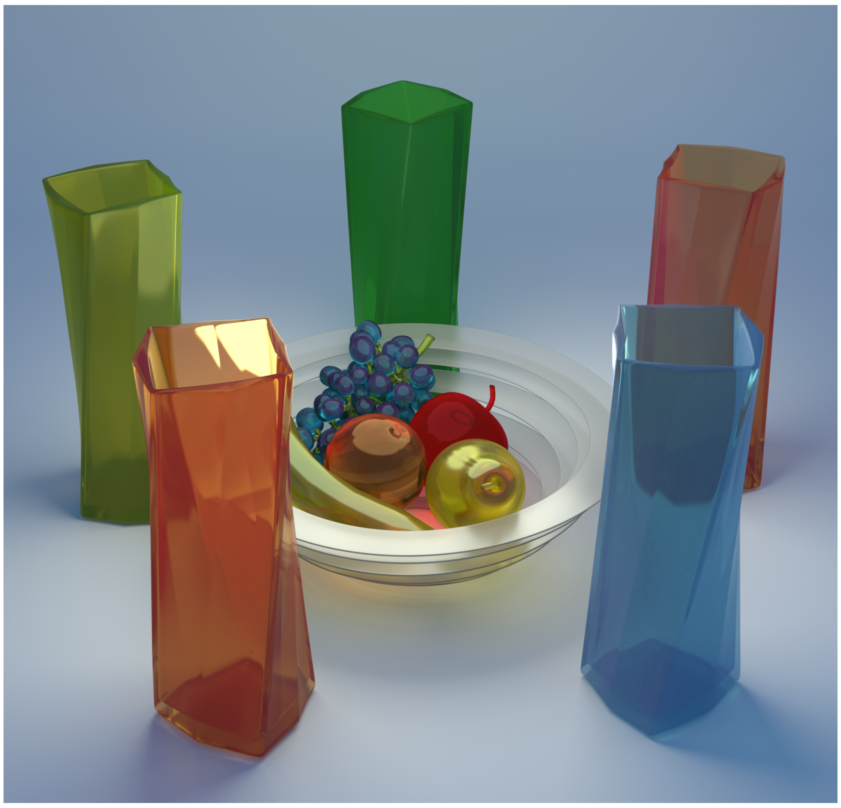


Figure 3.2: Glassy materials learned and synthesized by our technique using 150 training samples, 46 of which obtained a score greater than zero.

focus on providing tools for rapid mass-scale material synthesis to ease this process for novice and expert users alike – Figures 3.2 and 3.3 showcase example scenes where our technique was used to learn glassy and translucent materials. Instead of using a set of specialized shaders for each desired material class, it is generally possible to design one expressive shader that can represent a large swath of possible material models at the cost of increased complexity, which is often referred to as a “principled” or “uber” shader in the rendering community. Each parameterization of such a shader corresponds to one material model. Our strategy is to create a principled shader that is highly expressive, where the complexity downside is alleviated by the fact that the user never has to directly interact with it. To achieve this, we harness the power of three learning algorithms and show that this approach has several advantages compared to the classical workflow (i.e., direct interaction with a “principled” shader): when using our framework, the user is presented with a gallery where scores can be assigned to a set of proposed material models. These scores are used as training samples to adapt to these preferences and create new material recommendations. These recommendations are *controllable*, i.e., the user can choose the amount of desired variety in the output distribution, and our system retains the degree of physical correctness of its underlying shader. Normally, each of these new recommendations would have to be rendered via global illumination, leading to long waiting times. To alleviate this, we have replaced the renderer with a neural network that is able to predict these images in real time. We use a third learning algorithm to perform variant generation, which enables the user to fine-tune previously recommended materials to their liking in real time without requiring any domain expertise.

Furthermore, we explore combinations of these learning algorithms that offer useful real-time previews and color coding schemes to guide the user's attention to regions that are ample in variants with a high expected score and are also deemed *similar* to the fine-tuned input. We also show that our framework scales well with the number of sought materials and that it offers favorable modeling times compared to the classical workflow.

In summary, we present the following contributions:

- a framework for mass-scale material learning and recommendation that works with any high-dimensional principled shader,
- a Convolutional Neural Network to enable the visualization of the recommended materials in real time,
- a latent space variant generation technique that helps the user to intuitively fine-tune the recommended materials in real time,
- a novel way to combine all three learning algorithms to provide color coding for efficient latent-space exploration and real-time previews.

We provide our pre-trained neural network and the source code for the entirety of this project. This work also contains a supplementary video with a high-level overview of our system and a discussion of the results. We have also attached a compressed archive that contains the following materials: GPR training data and full workflows for the glassy, translucent, metals and minerals and glittery materials accompanied by 500 rendered images in the gallery that can be scored by the user, and a Blender scene containing the description of our principled shader.

3.3 Overview

The overall workflow of our system consists of two stages (see also Fig. 3.4)[2]. In the *first stage*, the user is presented with a gallery and asked to assign scores to the shown materials. After choosing a threshold value to control the variability of the output, a set of recommendations are computed (Section 3.4.1) and visualized via neural rendering

[2]In the Toy Tea Set scene, the volumetric absorption parameter for translucent materials is heavily scale-dependent: if the scale of the scene is larger than the one shown in the material preview scene (e.g., the teapots are typically larger), its increased optical thicknesses will result in darker outputs. We have slightly adjusted the absorptions to avoid this effect.

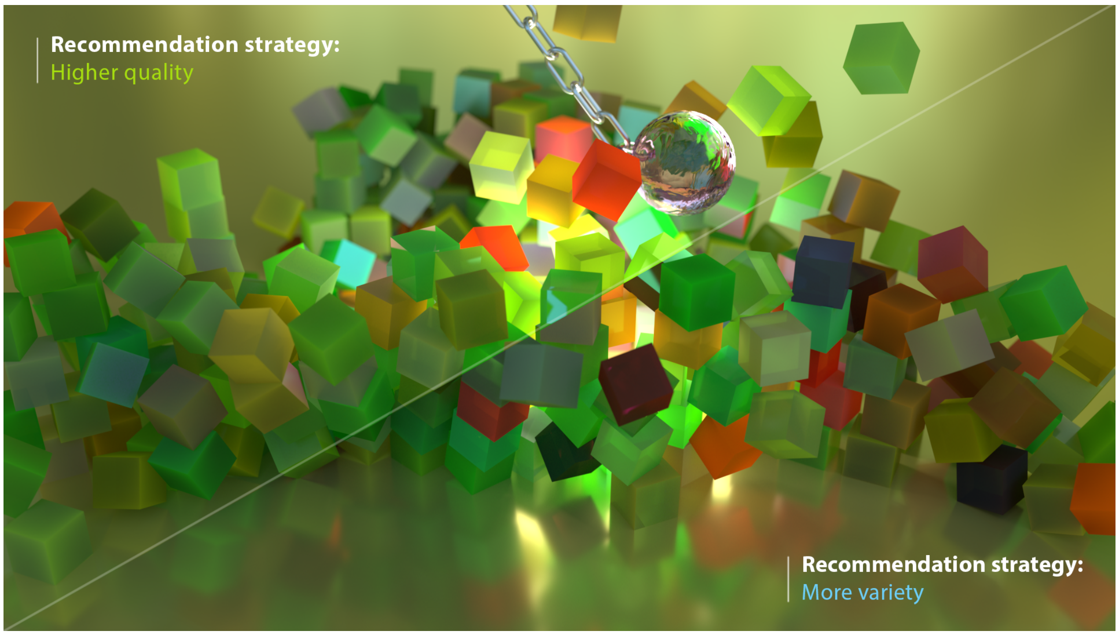


Figure 3.3: This scene was generated using our automatic workflow. The recommendations of our system are controllable, i.e., the user can easily adjust the recommendation thresholds to fine-tune the amount of variety in the output distribution.

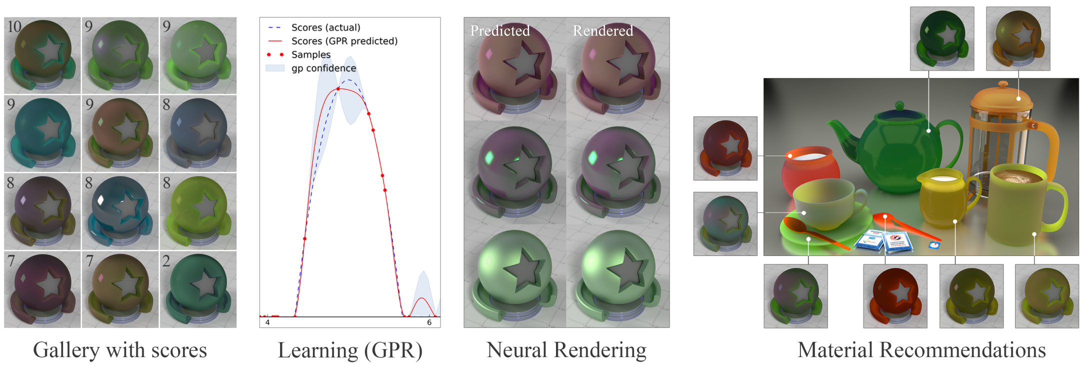


Figure 3.4: In the first step, the user is presented with a gallery and scores the shown materials according to their taste. Then, a regression is performed to obtain a preference function via Gaussian Process Regression, which can be efficiently sampled for arbitrarily many new material recommendations. These recommendations can be visualized in real time using our neural network in a way that closely resembles the images rendered with global illumination. In the final step, the recommended materials can be conveniently assigned to an existing scene.

(Section 3.4.2). The recommendations depend entirely on the user scores and may span multiple material classes. If the user wishes to fine-tune a subset of the recommended materials, a latent space is inferred for low-dimensional exploration (Section 3.4.3). The user then evaluates the result:

1. If the recommendations are acceptable – proceed to the next stage,
2. If the recommendations need refinement – assign scores to the *newly proposed gallery* or *adjust past rankings* and compute a new round of recommendations.

In the *second stage*, the user can choose from two ways to assign the recommended materials to a scene:

1. Automatic workflow – randomly assign the materials to the selected objects in the scene (Figures 3.1 (right) and 3.3). This is ideal for mass-scale material synthesis, when hundreds of materials are sought,
2. Assisted workflow – assign the materials to the scene manually and perform fine-tuning via *variant generation* (Section 3.5.2), with color coding (Section 3.5.1). This is ideal when up to a few tens of materials are sought and strict control is required over the output (Figures 3.1 (left) and 3.16).

The final scene with the newly assigned material models is then to be rendered offline. In Section 4.3 we first present the three learning algorithms, which can be used for the automatic workflow by themselves, while in Section 3.5 we show how to combine them to provide an interactive system.

3.4 Learning Algorithms for Material Synthesis

In this section, we outline the three main pillars of our system: **Gaussian Process Regression** to perform material learning and recommendation, a **Convolutional Neural Network** variant for real-time image prediction, and the **Gaussian Process Latent Variable Model** to embed our high-dimensional shader inputs into a 2D latent space to enable the fine-tuning of a select set of recommended materials (Fig. 3.5). Table 3.1 summarizes the notation used throughout this manuscript.

3.4.1 Material Learning and Recommendation

In this section, we propose a combination of Gaussian Process Regression, Automatic Relevance Determination and Resilient Backpropagation to efficiently perform material learning and recommendation. We also show that these recommendations are easily controllable.

Material learning. Gaussian Process Regression (GPR) is a kernel-based Bayesian regression technique that leverages prior knowledge to perform high-quality regression from a low number of samples[3]. It can be used to approximate a preference function $u(\mathbf{x})$ from a discrete set of n observations $\mathbf{U} = \left[u(\mathbf{x}_1), u(\mathbf{x}_2), \ldots, u(\mathbf{x}_n)\right]^T$, each of which can be imagined as point samples of a Gaussian where $\mathbf{x}_i \in \mathbb{R}^m$ encode the parameters that yield a BSDF model. We created a parameter space similar to Disney's "principled shader" [BS12] that comes in two versions: the $m = 19$ variant spans the most commonly used materials, i.e., a combination of diffuse, specular, glossy, transparent and translucent materials where the extended $m = 38$ version additionally supports procedurally textured albedos and displacements (see Section 5.11). A Gaussian Process is given by its mean and covariance $k(\mathbf{x}, \mathbf{x}')$ that describes the relation between individual material samples $\mathbf{x}$ and $\mathbf{x}'$. A squared exponential covariance function is given as

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left[-\frac{(\mathbf{x} - \mathbf{x}')^2}{2l^2}\right] + \beta^{-1}\delta_{xx'}, \tag{3.1}$$

with a given σ_f^2 variance and length scale l where $\beta^{-1}\delta_{xx'}$ is an additional noise term enabled by the Kronecker delta to yield a positive definite covariance matrix. This means that a highly correlated pair $\|\mathbf{x} - \mathbf{x}'\| \approx 0$ yields the maximum of the function, i.e., $k(\mathbf{x}, \mathbf{x}') \approx \sigma_f^2 + \beta^{-1}\delta_{xx'}$, leading to a smooth function approximation as $u(\mathbf{x}) \approx u(\mathbf{x}')$. Conversely, if $\|\mathbf{x} - \mathbf{x}'\|$ is large, the two observations show negligible correlation, therefore for a rapidly decaying covariance function, $k(\mathbf{x}, \mathbf{x}') \approx 0$.

The covariance matrix $\mathbf{K}$ is given by all possible combinations of the point samples

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & k(\mathbf{x}_1, \mathbf{x}_2) & \ldots & k(\mathbf{x}_1, \mathbf{x}_n) \\ k(\mathbf{x}_2, \mathbf{x}_1) & k(\mathbf{x}_2, \mathbf{x}_2) & \ldots & k(\mathbf{x}_2, \mathbf{x}_n) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_n, \mathbf{x}_1) & k(\mathbf{x}_n, \mathbf{x}_2) & \ldots & k(\mathbf{x}_n, \mathbf{x}_n) \end{bmatrix}, \tag{3.2}$$

the diagonal of $\mathbf{K}$ is therefore always $\sigma_f^2 + \beta^{-1}\delta_{xx'}$. The covariances for the unknown sample $\mathbf{x}^*$ are written as

$$\mathbf{k}_* = \left[k(\mathbf{x}^*, x_1), k(\mathbf{x}^*, x_2), \ldots, k(\mathbf{x}^*, x_n)\right]^T, \tag{3.3}$$

(where $x_i \in \mathbf{x}^*$) and $k_{**} = k(\mathbf{x}^*, \mathbf{x}^*)$. We define a zero-mean Gaussian Process over $\left[\mathbf{U}, u(\mathbf{x}^*)\right]^T$ with the covariance function $k(\mathbf{x}, \mathbf{x}')$:

$$\begin{bmatrix} \mathbf{U} \\ u(\mathbf{x}^*) \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} \mathbf{K} & \mathbf{k}_*^T \\ \mathbf{k}_* & k_{**} \end{bmatrix}\right). \tag{3.4}$$

[3]Throughout this manuscript, we will use the terms *samples* and *observations* interchangeably.

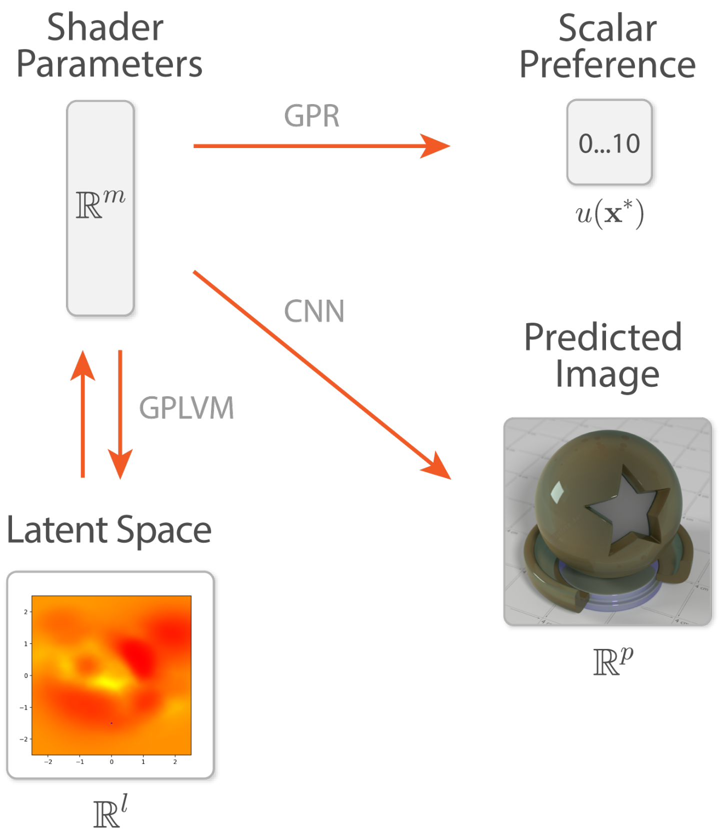


Figure 3.5: A high-level overview of our pipeline: GPR is used to learn the user-specified material preferences and recommend new materials which are subsequently visualized using our Convolutional Neural Network. Optionally, GPLVM can be used to provide an intuitive 2D space for variant generation.

We seek $u(\mathbf{x}^*)$ leaning on the knowledge that the conditional probability $P(u(\mathbf{x}^*) \,|\, \mathbf{U})$ follows a Gaussian distribution, therefore the closed-form solution for $u(\mathbf{x}^*)$ and its variance is obtained by

$$\begin{aligned} u(\mathbf{x}^*) &= \mathbf{k}_*^T \mathbf{K}^{-1} \mathbf{U}, \\ \sigma(u(\mathbf{x}^*)) &= k_{**} - \mathbf{k}_* \mathbf{K}^{-1} \mathbf{k}_*^T. \end{aligned} \tag{3.5}$$

The quality of the regression depends on the choice and the parameterization of the covariance function. If $\boldsymbol{\theta} = \{\sigma_f^2, l\}$ in (3.1) is chosen poorly, the result will suffer from severe over- or underfitting. To avoid this, a model selection step is performed to maximize the log-likelihood of the observed samples by choosing the appropriate hyperparameters, i.e.,

$$\log P(\mathbf{U}|\mathbf{x}, \boldsymbol{\theta}) = -\frac{1}{2}\mathbf{U}^T \mathbf{K}^{-1} \mathbf{U} - \frac{1}{2}\log|\mathbf{K}| - \frac{n}{2}\log 2\pi. \tag{3.6}$$

Symbol	Description	Type
$\mathbf{x}$	BSDF description	Vector
$u^*(\mathbf{x})$	Preference function (Ground truth)	Scalar
$u(\mathbf{x})$	Preference function (GPR prediction)	Scalar
n	Number of GPR samples	Scalar
$\mathbf{x}^*$	Unknown BSDF test input	Vector
$\mathbf{U}$	GPR training set	Matrix
m	Input BSDF dimensionality	Scalar
$k(\mathbf{x}, \mathbf{x}')$	Covariance function	Scalar
σ_f^2	Variance	Scalar
l	Length scale	Scalar
β^{-1}	Noise term	Scalar
$\delta_{xx'}$	Kronecker delta	Scalar
$\mathbf{K}$	GPR covariance matrix	Matrix
$\boldsymbol{\theta}$	Covariance function parameterization	Vector
$\boldsymbol{\alpha}$	Learning rate	Vector
$\phi(\mathbf{x}^*)$	CNN image prediction of a BSDF	Matrix
$\mathbf{X}$	GPLVM training set	Matrix
$\mathbf{L}$	Low dimensional latent descriptor	Matrix
l	Latent space dimensionality	Scalar
$\mathcal{N}(x\,\vert\,\mu, \sigma^2)$	$\frac{1}{\sqrt{2\pi\sigma^2}}\exp(-\frac{(x-\mu)^2}{2\sigma^2})$	Scalar
z	Number of GPLVM samples	Scalar
$\mathbf{K}'$	GPLVM covariance matrix	Matrix
$\psi(\mathbf{l}^*)$	Mapping from latent to observed space	Vector
$m(\mathbf{x})$	$\frac{1}{2}(u(\mathbf{x}) + u^*(\mathbf{x}))$	Scalar
τ	Recommendation threshold	Scalar
r	Grid resolution	Scalar
$s(\mathbf{x}^*, \mathbf{x}')$	Similarity (CNN prediction)	Scalar
$u(\mathbf{x}')$	Preference score (GPR prediction)	Scalar

Table 3.1: Notation used throughout this work, in order of occurrence.

The derivatives of the log-likelihood with respect to the hyperparameters and a learning rate $\boldsymbol{\alpha}$ are given by

$$\frac{\partial}{\partial\theta_j}\log P\Big(\mathbf{U}|\mathbf{x}, \boldsymbol{\theta}\Big) = -\frac{1}{2}\,\mathrm{tr}\left\{\Big(\boldsymbol{\alpha}\boldsymbol{\alpha}^T\mathbf{K}^{-1}\Big)\frac{\partial\mathbf{K}}{\partial\theta_j}\right\}, \tag{3.7}$$

which can be used with gradient-based optimization techniques. In some cases, the user is looking for a class of material models where, for instance, the surface reflectance properties are of great importance and the choice of albedos is irrelevant (e.g., carpaint material variants). In this case, using one fixed length scale for all features leads to poor predictions. To this end, we have also used Automatic Relevance Determination [Mac96, Nea12] and assigned a $\boldsymbol{\theta}$ to each dimension (with appropriate modifications to (3.1)) to open up the possibility of discarding potentially irrelevant features. However, this substantially increases the dimensionality of the optimization problem, to a point where classical methods such as L-BFGS-B [BLNZ95] and the Scaled Conjugate Gradient method [Møl93] prove to be ineffective: lower learning rates are theoretically able to find the desired minima, but slow down considerably in shallow regions while larger learning rates introduce oscillation near the minima. To this end, instead of using a fixed step size that is proportional to the local gradient, we adapt the learning rate based on the topology of the error function by using Resilient Backpropagation [RB92, BR13], a technique originally designed for training neural networks. This significantly reduces the number of required learning steps and has always succeeded finding usable minima in our experiments.

Material recommendation. Given enough learning samples, u will resemble the true user preferences, therefore high-scoring material recommendations can be obtained by simply rejection-sampling it against a minimum acceptable score threshold. The rejection rates depend on the properties of u: choosing a high threshold will result in high-quality recommendations at the cost of higher rejection rates and decreased variety. Conversely, a larger variety of recommendations can be enforced by lowering the target threshold. Upon encountering a rejection ratio that is unacceptably high, we extend the rejection sampler with a low number of stochastic hillclimbing steps. This offers a considerably improved ratio at the cost of a minimal increase in sample correlation. Generally, we have found setting the recommendation threshold between 40% and 70% of the maximum possible score to be a good tradeoff between sample quality and variety (Fig. 3.3, higher quality (left) – 70%, more variety (right) – 40%).

3.4.2 Neural Networks and Rendering

After the learning and recommendation step have taken place, a gallery is generated with a prescribed amount of recommendations. The GPR and the recommendation steps take only a few seconds (Table 3.3), however, rendering all 300 recommendations (a typical number throughout our experiments) would take over 4 hours, which is a prohibitively long time for practical use. To cut down the time between the two steps, we introduce a neural network-based solution. Deep neural networks are universal function approximators that have proven to be remarkably useful for classification and regression problems. Typically, the dimensionality of the input is orders of magnitude larger than that of the output (e.g., image classification). Convolutional Neural Networks [LBBH98] (CNNs) excel at solving problems of this form; their advantages include augmenting neurons with a receptive field to take advantage of the locality of information in images and their pooling operations that reduce the number of parameters in each layer. In this

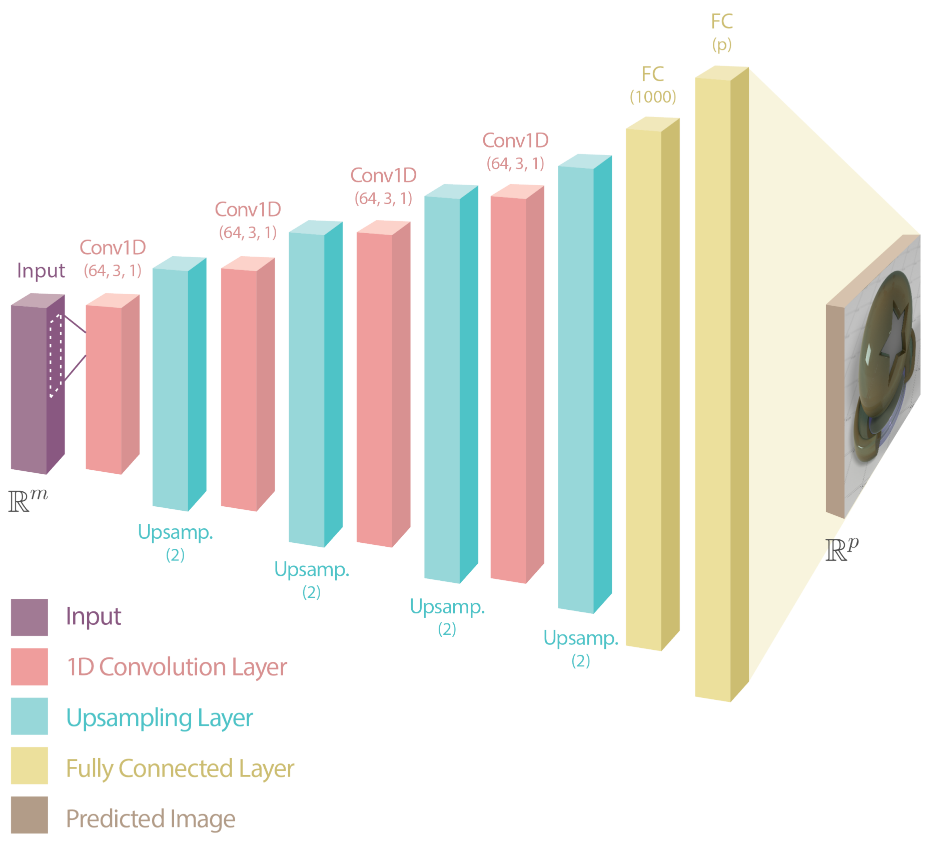


Figure 3.6: Our decoder network takes the shader description as an input and predicts its appearance. Due to the fact that we have an atypical problem where the input shader dimensionality is orders of magnitude smaller than the output, the input signal is subjected to a series of 1D convolution and upsampling steps.

work, we are concerned with an atypical adjoint problem where images are to be predicted pixel by pixel from the shader input, $\phi \colon \mathbb{R}^m \to \mathbb{R}^p$, where the input dimensionality m is in the order of tens, which is to be mapped to an output of p dimensions, which represents the largest possible output that fits into the GPU memory along with its decoder network, i.e., in our case, a 410^2 image with three color channels. We will refer to problems of this kind as *neural rendering.*

Neural rendering remains an unsolved problem in general. However, as our case is constrained to material modeling, the geometry and lighting setups can be kept constant, therefore it is possible to create a sufficiently deep network and training set to predict images that are nearly indistinguishable from the ones rendered until convergence with full global illumination. Using deconvolutional layers [ZKTF10, NHH15] would be a natural choice for ϕ, however, as few of the most common software packages support it and unwanted artifacts may appear during image generation [ODO16], instead, the input

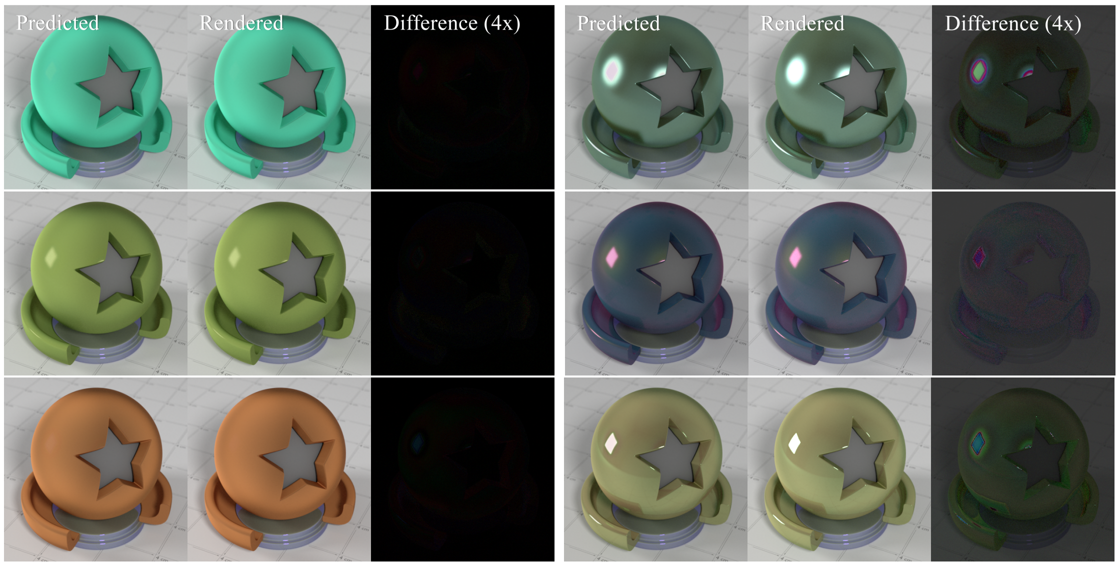


Figure 3.7: The best (left side) and worst-case (right side) predictions by our neural network on a set of 250 images. Mean PSNR: 37.96dB, minimum: 26.05dB, maximum: 48.70dB.

signal is subjected to a series of 1D convolutions [4], each followed by an upsampling layer to inflate the number of parameters as we advance deeper into the network (Fig. 3.8).

This architecture is similar to the decoder part of Convolutional Autoencoders [MMCS11] and can be described with the shorthand notation of 4x$\{$Conv1D$(64, 3, 1)$ – Upsampling$(2)\}$ – FC(1000) – FC$(410^2 \cdot 3)$, where the parameters of the convolutional layer are *number of filters*, *spatial kernel size* and *strides*, respectively. These layers use exponential linear units [CUH15] with Glorot-initialization [GB10] and are trained via the Adam optimizer [KB14] ($lr{=}10^{-3}, \beta_1{=}0.9, \beta_2{=}0.999, \epsilon{=}10^{-8}$, decay$=0$). Normally, a network of this size introduces severe overfitting, even in the presence of $\mathcal{L}_1/\mathcal{L}_2$ regularization [NH92, ZH05] or dropout [SHK+14], especially if learning takes place on a given, fixed dataset. However, as we can create our own dataset, i.e., a potentially infinite number of shader-image pairs via rendering, we are able to evade this problem by creating a sufficiently large training set. In the interest of efficiency, we have generated 45000 LDR shader-image pairs with a spatial resolution of 410^2 and 250 samples per pixel over 4 weeks on a consumer system with a NVIDIA GeForce GTX TITAN X GPU, and since no means were required to prevent overfitting, our training process converged to 10^{-2} (RMSE), a negligible but non-zero $\mathcal{L}_2$ training and validation loss (where a zero loss would mean containing all the noise from the training set) in less than 30 hours. Our validation set contained 2500 images and the measured losses correlated well with

[4]The input of the network is a set of shader parameters that bear some locality, as e.g., R,G,B albedos for a material node are often adjacent. However, they have no meaningful 2D spatial relation to each other, therefore 1D convolutions are preferable.

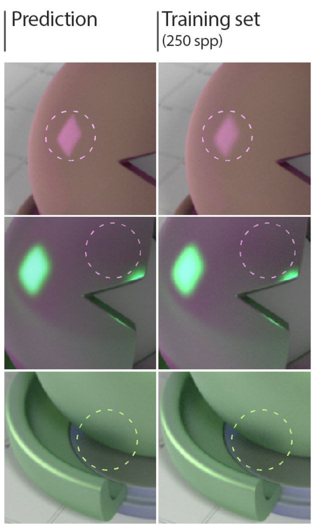


Figure 3.8: An additional desirable property of our proposed neural network is that it contains just enough layers to infer the most important features, but not enough to represent the high-frequency noise in the dataset.

real-world performance. This pre-trained network can be reused as long as the shader description remains unchanged and the inference of a new image typically takes 3 to 4 milliseconds. A further advantage of this architecture is that similarly to Denoising Autoencoders [VLL+10], it also performs *denoising* on the training samples. This noise filtering property of the neural network is subject to a tradeoff – adding more layers leads to more faithfully rendered images at the risk of additionally fitting the noise in the dataset. A more principled approach could be developed by using modern neural network visualization techniques to observe the amount of noise contained within the filters [OSJ+18].

A key observation is that the more layers the neural network contains, the more high-frequency details it will be able to represent (a similar effect has been observed in Fig 4., Saito et al. [SWH+16]). This means that our proposed neural network contains enough layers to capture the important features to maximize visual quality, but not enough to learn the high-frequency noise contained in the dataset (Fig. 3.8). Normally, this denoising process requires the presence of auxiliary feature buffers and longer computation times [SD12] or other deep learning approaches using more complex architectures [BVM+17]. In our case, this step does not require any additional complexity and is inherent to the structure of our network. Beyond producing less noisy images in real time, this is also a significant advantage in easing the computational load of creating new datasets as the training images do not have to be rendered until convergence. We took advantage of this by using only 250 samples per pixel for each training image, which took six times less than the standard 1500 samples that would be required for perfectly converged training samples, cutting down the total time to produce the training set from 24 to 4 weeks.

Initially, instead of synthesizing the corresponding images for new recommended sample points, using the images from the training database instead appears to be a reasonable optimization option. The issue of this approach is that it is struck with the curse of dimensionality; if one generates n training points on $[0,1]^m$ with uniform distribution, and the average $\mathcal{L}_2$ distance between a new random query point and the closest database match is 0.71 when $m = 19, n = 17 \cdot 10^4$ (simpler shader) and 1.45 for $m = 38, n = 17 \cdot 10^4$ (extended shader). This difference is significant as some shader parameters are non-linear and a difference of 10^{-1} in the index of refraction or glass roughness leads to a drastically different material appearance or non-physical results. We note that we obtained these results via a Monte Carlo simulation as a complete proof of this result would be overly verbose (e.g., the average distance is close to 0.52 for $m = 2, n = 1$ [MMP99, BP09]).

3.4.3 Latent Space Variant Generation

After being presented with a gallery of material models, the user may find that some recommendations are close to their preference, but some require fine-tuning to fit their artistic vision for a scene. Adjusting the materials by hand requires domain expertise and a non-trivial amount of trial and error. In our solution, we seek a dimensionality reduction technique that maps the shader inputs into a 2D latent space where similar materials can be intuitively explored. Non-linear manifold learning techniques come as a natural choice for this kind of problem, however, the most well-known methods [BN03, MH08, TDSL00] could not find coherent structures in 2D. Beyond that, these techniques are unable to interpolate between the embedded samples, therefore they would only be useful for visualization in a latent space, but not for exploration. Dimensionality reduction with autoencoders [HS06] would require orders of magnitude more samples to work properly (we have at most a few tens at our disposal), which is infeasible as it would be too laborious for the user to provide that many scores through the presented gallery.

The Gaussian Process Latent Variable Model (GPLVM) [Law04] is a non-linear dimensionality reduction technique which is able to embed a few tens of high-scoring materials from the gallery $\mathbf{X} = \left[\cdots \mathbf{x}_i \cdots\right]^T$ with $\mathbf{x}_i \in \mathbb{R}^m$ into a set of low dimensional latent descriptors $\mathbf{L} = \left[\cdots \mathbf{l}_i \cdots\right]^T$ with $\mathbf{l}_i \in \mathbb{R}^l$. Typically, $m \gg l$, in our case, $m = 19$ and $l = 2$ to make sure that variant generation can take place conveniently on a 2D plane. The likelihood of the high-dimensional data using z high-scoring training samples is given as

$$P\Big(\mathbf{X}|\mathbf{L},\boldsymbol{\theta}\Big) = \prod_{i=1}^{z} \mathcal{N}\Big(\mathbf{X}_i|\,\mathbf{0},\ \mathbf{K}'+\beta^{-1}\mathbf{I}\Big), \tag{3.8}$$

where $\mathbf{K}'$ is a covariance matrix similar to $\mathbf{K}$ containing elements akin to (3.1) with a substitution of $k(\mathbf{x},\mathbf{x}') \to k(\mathbf{l},\mathbf{l}')$. In this case, the optimization takes place jointly over the latent values in $\mathbf{L}$ and $\boldsymbol{\theta}$, i.e.,

$$\mathbf{L}^*,\boldsymbol{\theta}^* = \underset{\mathbf{L},\boldsymbol{\theta}}{\operatorname{argmax}} \log\Big[P\Big(\mathbf{X}|\mathbf{L},\boldsymbol{\theta}\Big)\Big]. \tag{3.9}$$

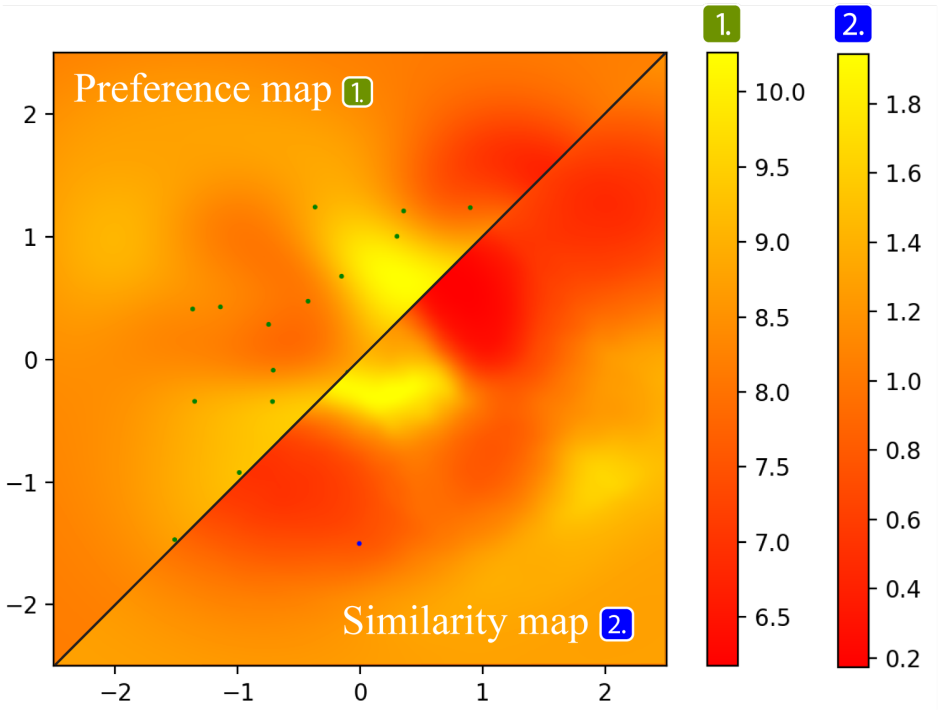


Figure 3.9: Endowing the latent space with the expected preferences (upper left) and similarities (lower right). The green dots represent the embedded training samples, where the blue dot shows the reference input material to be fine-tuned.

Even though PCA disregards the non-linear behavior of BSDF parameters [LFTG97], it serves as a formidable initial guess for $\mathbf{L}$ [Law04] and $\boldsymbol{\theta}$ is initialized with a wide prior. Beyond the ability to learn efficiently from a handful a samples, a further advantage of this method is that a new mapping can be made between the latent and observed space $\mathbf{x}^* = \psi(\mathbf{l}^*)$, i.e.,

$$\begin{bmatrix} \mathbf{X} \\ \mathbf{x}^* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} \mathbf{K}' & \mathbf{k}_*'^T \\ \mathbf{k}_*' & \mathbf{k}_{**}' \end{bmatrix}\right), \tag{3.10}$$

yielding the final closed-form solution

$$\begin{aligned} \psi(\mathbf{l}^*) &= \mathbf{k}_*'^T \mathbf{K}'^{-1} \mathbf{X}, \\ \sigma(\psi(\mathbf{l}^*)) &= \mathbf{k}_{**}' - \mathbf{k}_*' \mathbf{K}'^{-1} \mathbf{k}_*'^T. \end{aligned} \tag{3.11}$$

By storing and reusing $\mathbf{K}'^{-1}$, this mapping can be done in negligible time, which allows the user to rapidly generate new material variants in this 2D latent space.

3.5 Interactive Latent Space Exploration

We have introduced a system using three learning algorithms in isolation. GPR enables material learning and recommendation from a few learning samples, the CNN opens up the possibility of neural rendering, and the high-dimensional shader can be non-linearly embedded in a 2D latent space using GPLVM. In this section, we propose novel ways to combine these algorithms to obtain a system for rapid mass-scale material synthesis and variant generation. A more rigorous description of the final algorithm is presented in Section 3.5.3.

3.5.1 GPLVM Color Coding

When using GPLVM, each point in the 2D latent space corresponds to an m-dimensional vector that describes a material model. Since we have used GPR to learn the correspondence between these materials and user preferences, it is possible to combine these two techniques to obtain the expected scores for these samples. This combination enables a useful visualization of the latent space where these expected preferences appear in the form of color coding. This *preference coding* is useful to highlight regions of the latent space that encode favorable materials, however, when fine-tuning a chosen material, the requirement of obtaining *similar* materials is of equal importance. Since our CNN is able to predict images in real time, we propose subdividing the latent space into a 2D grid, where an image can be predicted in each gridpoint. This image can be compared to the image of the material we wish to fine-tune via a distance metric of choice (e.g., $\mathcal{L}_1/\mathcal{L}_2$), therefore, the latent space can thus be endowed with *similarity information* as well.

The preference map is global, i.e., it remains the same regardless of our input material where the similarity map depends on our current material that is used as a starting point (Fig. 3.9). The product of these two maps offers an effective way to create variants of a source material that are similar, and are highly preferred according to the learned preferences.

3.5.2 Real-Time Variant Generation

Exploring the 2D latent space of the learned materials is of limited usefulness when the user has to wait for 40-60 seconds for each new image to be rendered. A typical use case involves sweeping motions that require near instantaneous feedback from the program. As the 2D output of the GPLVM can be projected back to the high-dimensional material space (with a dimensionality increase of $l \to m$), this output can be combined with our CNN, which provides real-time image predictions to allow efficient exploration in the latent space with immediate feedback. We demonstrate several such workflows in our supplementary video.

3.5.3 Pseudocode and Shader Description

We provide the full pseudocode of our system in Algorithms 1-3. Note that the symbol "_" in lines $7, 14$ and 16 refer to throwing away part of the function return value (i.e., using only the first dimension of the output of a 2D function).

Algorithm 1 Gaussian Material Synthesis

```
 1: given $\tau, r$                                   ▷ Recommendation threshold, grid resolution
 2: for $i \leftarrow 1$ to $k$ do                     ▷ $k$ GPR training samples
 3:     Generate random BSDF $\mathbf{x}$
 4:     $\mathbf{U}_i \leftarrow u(\mathbf{x})$
 5: $\mathbf{X} \leftarrow \{\mathbf{x} \in \mathbf{U} \mid \mathbf{x} > \tau\}$     ▷ GPLVM training set
 6: for $i \leftarrow 1$ to $l$ do                     ▷ Recommendations
 7:     while Score$(\mathbf{X}, \mathbf{x}, \beta^{-1}) < \tau, \_$ do
 8:         Generate random $\mathbf{x}$
 9:     $\mathbf{R}_l \leftarrow \mathbf{x}$
10: Choose $\mathbf{x}^* \in \mathbf{R}$ for variant generation
11: Display $\phi(\mathbf{x}^*)$
12: for $i, j \leftarrow 1$ to $r$ do                  ▷ GPLVM color coding, Section 3.5.1
13:     init $\mathbf{G}_{ij}$ gridpoint with coordinates $i, j$ and resolution $r$
14:     $\mathbf{x}', \_ \leftarrow$ Latent $(\mathbf{X}, \mathbf{G}_{ij}, 2, \beta^{-1})$
15:     $s(\mathbf{x}^*, \mathbf{x}') \leftarrow ||\phi(\mathbf{x}^*) - \phi(\mathbf{x}')||_{\mathcal{L}_2}$     ▷ Similarity (CNN)
16:     $u(\mathbf{x}'), \_ \leftarrow$ Score$(\mathbf{U}, \mathbf{x}', \beta^{-1})$     ▷ Preference (GPR)
17:     $T_{ij} \leftarrow s(\mathbf{x}^*, \mathbf{x}') u(\mathbf{x}')$     ▷ Store product coding
18:     assign color $\mathbf{T}_{ij}$ to gridpoint $\mathbf{G}_{ij}$
19: while Given user displacement $\boldsymbol{\delta}$ do     ▷ Explore latent space
20:     Display $\phi(\mathbf{x}^* + \boldsymbol{\delta})$
```

Algorithm 2 Scoring a new BSDF

```
1: function Score$(\mathbf{U}, \mathbf{x}^*, \beta^{-1})$     ▷ Score new BSDF, Section 3.4.1
2:     init $\mathbf{K}, \boldsymbol{\theta}$
3:     $\boldsymbol{\theta}^* \leftarrow \text{argmax}_{\boldsymbol{\theta}} \log [P(\mathbf{U}|\mathbf{x}, \boldsymbol{\theta})]$
4:     $u(\mathbf{x}^*) \leftarrow \mathbf{k}_*^T \mathbf{K}^{-1} \mathbf{U}$
5:     $\sigma(u(\mathbf{x}^*)) \leftarrow k_{**} - \mathbf{k}_* \mathbf{K}^{-1} \mathbf{k}_*^T$
6:     return $u(\mathbf{x}^*)$, $\sigma(u(\mathbf{x}^*))$
```

Our shader is defined as a combination of a set of base BSDFs and mix shaders that return a linear interpolation of two inputs. In the following description, the numerical ID, BSDF model names and their parameters are enumerated.
1: *Diffuse BSDF* (r,g,b albedos), **2**: *Beckmann Glossy BSDF* (r,g,b albedos, roughness), **3**: *Mix shader* (connect 1,2), **4**: *Beckmann Glass BSDF* (r,g,b albedos, roughness, IOR), **5**: *Translucent BSDF* (r,g,b albedos), **6**: *Mix shader* (connect 4,5), **7**: *Mix shader*

Algorithm 3 Latent space mapping

```
1: function LATENT($\mathbf{X}, \mathbf{x}^*, l, \beta^{-1}$)          ▷ Latent mapping, Section 3.4.3
2:     given $\mathbf{x}^* = \psi(\mathbf{l}^*)$
3:     init $\mathbf{K}^{'}, \boldsymbol{\theta}$
4:     $\mathbf{L}^*, \boldsymbol{\theta}^* \leftarrow \text{argmax}_{\mathbf{L},\boldsymbol{\theta}} \log \Big[P\Big(\mathbf{X}|\mathbf{L},\boldsymbol{\theta}\Big)\Big]$
5:     $\psi(\mathbf{l}^*) \leftarrow \mathbf{k}_*^{'T}\mathbf{K}^{'-1}\mathbf{X}$
6:     $\sigma(\psi(\mathbf{l}^*)) \leftarrow \mathbf{k}^{'}_{**} - \mathbf{k}^{'}_*\mathbf{K}^{'-1}\mathbf{k}_*^{'T}$
7:     return $\psi(\mathbf{l}^*)$, $\sigma(\psi(\mathbf{l}^*))$
```

Table 3.2: All three optimization techniques produce competitive JSD values in the lower dimensional case (i.e., $m = 19$). In the case of the extended shader ($m = 38$), RProp consistently outperforms L-BFGS-B and SCG regardless of the number of training samples (n).

Scene	m	n	RProp	L-BFGS-B	SCG
Glassy	19	150	**0.08**	0.10	**0.08**
Glassy	19	250	0.09	0.09	**0.08**
Glassy	19	500	**0.07**	**0.07**	**0.07**
Translucent	19	150	**0.17**	0.18	0.18
Translucent	19	250	**0.19**	**0.19**	–
Translucent	19	500	**0.17**	**0.17**	**0.17**
Glassy	38	150	**0.41**	0.58	0.58
Glassy	38	250	**0.35**	0.57	0.57
Glassy	38	500	**0.14**	0.53	0.38
Metals/Minerals	38	150	**0.53**	0.55	0.55
Metals/Minerals	38	250	**0.44**	0.52	–
Metals/Minerals	38	500	**0.32**	0.62	0.60

(connect 3,6).
The volume absorption for the Glass and Translucent BSDFs are inherited (and shared) from a separate node. The extended shader contains a combination of several noise models, a similar mixing logic and individual weighting factors to control. In the interest of simplicity, we provide a visual description of this shader in Fig. 3.12.

3.6 Results

In this section, we evaluate the three learning algorithms in isolation and demonstrate the utility of our whole system by recording modeling timings against the classical workflow for three practical scenarios. Furthermore, we also discuss how the proposed system handles our extended shader.

Learning material spaces. After having obtained u, we are interested in measuring the quality of the regression by relating it to the true user preference function u^*. By normalizing both functions and treating them as probability distributions, the Jensen-Shannon divergence (JSD) yields a suitable metric to distinguish how much information is lost if u is used as a proxy for the unknown u^*, i.e.,

$$\begin{aligned}&\mathrm{JSD}\big(u(\mathbf{x})\,||\,u^*(\mathbf{x})\big) = \\ &\quad \frac{1}{2}\int_{-\infty}^{+\infty} u(\mathbf{x}) \log \frac{u(\mathbf{x})}{m(\mathbf{x})}\, d\mathbf{x} + \frac{1}{2}\int_{-\infty}^{+\infty} u^*(\mathbf{x}) \log \frac{u^*(\mathbf{x})}{m(\mathbf{x})}\, d\mathbf{x},\end{aligned} \tag{3.12}$$

where $m(\mathbf{x}) = \frac{1}{2}\big(u(\mathbf{x}) + u^*(\mathbf{x})\big)$. We have recorded the JSD produced by minimizing (3.7) with RProp, L-BFGS-B and the Scaled Conjugate Gradient method and found that all three techniques are competitive for the lower-dimensional case, i.e., $m = 19$. In the case of the extended shader, RProp consistently outperformed L-BFGS-B and SCG, both of which often got stuck in poor local minima even when being rerun from many randomized initial guesses (Table 3.2). For the high-dimensional cases with over 200 training samples, SCG did not always converge despite a non-singular K due to round-off errors.

We have used two challenging cases to demonstrate the utility of our system by learning the material space of glassy and translucent materials. These cases are considered challenging in a sense that these materials are relatively unlikely to appear via random sampling: in the glassy use case, 81% of the samples in the initial gallery were scored zero. This ratio was 90% for the translucent case. This means that the recommender system has to learn the appropriate sample distribution from a modest number of non-zero data points. We have scored 1000 glassy and translucent materials on a scale of 0 to 10 to use as a ground truth dataset, where the first 250 samples were used as training data for the GPR. In each case, our technique was able to generate high-quality recommendations from 46 (glassy) and 23 (translucent) non-zero observations. The remaining 750 samples were used for cross-validation to compute a reliable estimate of the JSD. In both cases, the training took 7.22s and as a result, an arbitrarily large gallery of recommendations can be generated in 0.06s per recommendation on a mid-range consumer Intel Core i5-6600 CPU (see Table 3.3 for a detailed breakdown). The metals and minerals scene in Fig. 3.10 showcases a *multi-round* learning scenario where the recommendation gallery was scored and re-used to generate a second, more relevant gallery of materials (all other cases use one round of scores)[5].

Human biases. We have identified several recurring biases throughout our experiments. For instance, when the gallery is presented as a 2D grid, the score of a material often depends on its surroundings, e.g., it is typically rated higher when the material is

[5]If the recommendations are in line with the user's artistic vision, but require fine-tuning, assigning scores to the newly created gallery and using them as training data for one more round is expected to improve the relevance of the recommendations (we used a two-round scheme for metals and minerals). Since these new samples are *concatenated* to the previous scores, it is advisable to first make sure that the scores from the first round do not contain many conflicting decisions. If the recommendations are not at all acceptable, it is advisable to revisit the initial rankings.

surrounded by unfavorable examples. Users are also typically more susceptible to assign a high score to a mediocre sample early in the process before they have seen the best matches the shader has to offer. The perception of different light simulation effects may also introduce imperfections in the scoring process, e.g., the effect of translucency is difficult to recognize in moderation, and therefore there are barely any mid-scoring samples, with most scores being either extremely low or high. In Fig. 3.17, we show that due to these biases, the Jensen-Shannon Divergence does not recede to zero as we add more training samples, and that despite these distortions, it is still possible to perform regression and material recommendations of acceptable quality.

Variant generation. A synthesized material of choice can be fine-tuned via variant generation in our 2D latent space. The exploration is guided by two different kinds of color-coding and real-time previews of the final materials. We demonstrate the usefulness of this element of our system in a real-world scenario by reducing the vividness of the grape material in the glassy still life scene (Fig. 3.16 and supplementary video). The color-coded regions denote outputs that are preferred *and* similar to the input material and form an island that is easy to explore. The preference and similarity maps are computed on a 50^2 and 20^2 grid respectively using an NVIDIA GeForce GTX TITAN X GPU and are subjected to bilinear interpolation in our visualizations.

Neural rendering comparisons. Because of our restricted problem definition, our neural network can mimic a global illumination renderer with high-quality predictions, and does not require retraining when combined with a sufficiently expressive principled shader. Three average case predictions are shown in Fig. 3.4 (middle). Three of the best- and worst-case predictions and their difference images as well as PSNR values are reported on a set of 250 images in Fig. 3.7. Querying this neural network takes 3-4ms on average and in every case, the predicted images were close to indistinguishable from the ground truth. An additional advantage our proposed architecture is that even though the training set contained moderately noisy images (250 spp, visible in the floating image in Section 3.4.2 and zooming in to the “Gallery with scores” part of Fig. 3.4), the predicted images appear smoother. We have also computed the PSNR error values against the ground truth, and have found that they were comparable, i.e., 37.9dB for the predictions and 40.2dB for the training set when compared against the ground truth. In our work, we argue that the visual quality of the predictions is higher because of the denoising property of the neural network. The measured PSNR is slightly lowered by the fact that some predicted images are off by a few points of brightness value; these are distributed uniformly over the image and are hence imperceptible for the user.

Modeling and execution time. To show that our system is useful for novice and expert users alike, we recorded the time required to model 1, 10, and 100 similar materials using Disney’s “principled” shader [BS12] against our technique. The two main user types to be compared to is a novice user who has no knowledge of light transport and material modeling, and an expert with significant experience in material modeling. Both were

Stage	Time [s]	Size
GPR	7.22	250
Recommendation	0.06 / 17.4	1 / 300
GPLVM	1.96	16
CNN	0.04	410^2
Preference coding	2.75	50^2
Similarity coding	8.15	20^2
Sum	20.18 / 37.52	

Table 3.3: Execution times for different stages of the proposed method. The 'Size' column stands for the size of the problem at hand, i.e., training samples for GPR and GPLVM, number of recommended materials, image resolution for the CNN, and 2D spatial resolutions for preference and similarity coding.

allowed several minutes to experiment with the principled shader before starting. The novice and expert users took 161s and 52s to obtain one prescribed base material model (a slightly scattering blue glass material with a small, non-zero roughness). Creating subsequent variants of this material took 29s and 19s where most of the time was spent waiting for a reasonably converged rendered image to show the minute differences between the base material and the new variant. Using our technique, in the presented gallery, it took an average of 2s to score a non-zero sample and 0.4s for a sample with the score zero. Typically, in our workflows, 250 observations were used to learn the material spaces and provide recommendations therefore we based our timings on that number. When only one material model is sought, novice users experience roughly equivalent modeling times when using our proposed system. In the case of mass-scale material synthesis, modeling times with our system outperform expert users. Beyond cutting down the time spent with material modeling, our system provides several other advantages over the traditional workflow: it does not require any domain expertise, provides real-time denoised previews throughout the process, and during scoring, the users are exposed to a wider variety of examples. This last advantage is especially useful for novices who do not necessarily have a prior artistic vision and are looking for inspiration. We also note that the workflow timings are often even more favorable as 150 samples are enough to provide satisfactory results for learning challenging material spaces (Fig. 3.17). Our intention in Fig. 3.18 was to show that our timings are appealing even in the more pessimistic cases. Furthermore, we have found that the fixed cost of the direct interaction with a principled shader is consistent among users. Our expert and novice users noted that most of their time was spent waiting for noise to clear up in the rendered images when a parameter is changed. This effect is particularly pronounced during variant generation, where the user has to wait until the minute differences between the old and new variant are revealed. This is a shortcoming that is inherent in the fact that all images have to be re-rendered and remains true for all users.

Adding displacements. In the results shown so far, we have used a shader with $m = 19$ as a basis for the training process, however, the GPR and GPLVM steps are capable of learning significantly higher-dimensional inputs. To demonstrate this, we have created a higher-dimensional SVBRDF shader that includes procedural textures and displacements. This shader ($m = 38$) is even more in line with our design principles, i.e., more expressive at the cost of being less intuitive, which is alleviated by using learning methods instead of interacting with it directly. In this case, material recommendation and learning steps still perform well – we have used 500 samples to learn metals and minerals and 150 samples for glittery Christmas ornament materials (Fig. 3.10). The limitation of this extended shader is the fact that it is ample in localized high-frequency details that our CNN was unable to represent. We note that this is a hardware limitation, and by adding more layers, more of these details are expected to appear, our architecture will therefore be able to predict these features as further hardware advancements take place, noting that this may require a higher sampling rate for the training set. After the recommendation and material assignment steps, displacements can be easily added by hand to the simpler shader setup (as shown in the supplementary video).

3.7 Future Work

Our technique starts out by showing a randomly generated gallery to the user to obtain a set of scores. These BSDFs are sampled with uniform distribution. We have experimented with improving it with an active learning scheme [KGUD07] to introduce adaptivity to the sampling process, making the newer gallery elements more relevant as they depend on the user-specified scores. For instance, the sampler can be equipped with novelty search [LS11, LS08] to aggressively look for unexplored regions that may be preferred by the user. Furthermore, after obtaining the first few samples, quick GPR runs can take place, and instead of standard uniform distribution, the upcoming gallery images can be drawn from this learned intermediate distribution. We have found this scheme highly effective, and in the supplementary materials, we provide a case with a "two-round" recommendation run for metals and minerals. Novice users can be further aided through automatic variant generation of new material(s) $\mathbf{x}'$ to fine-tune an input $\mathbf{x}^*$ by maximizing $\lambda s(\mathbf{x}^*, \mathbf{x}') + (1 - \lambda)u(\mathbf{x}')$ for a multitude of different $\lambda \in (0, 1)$ choices instead of relying solely on maximizing $u(\mathbf{x}')$ for recommendations (see Table 3.1 for details on the notation). By using GPR and GPLVM, not only the regressed outputs, but also their confidence values can be visualized (Equations (3.5) and (3.11)), or used as additional information for active learning. High-dimensional measured BRDF representations may also be inserted into the system. Due to the increased parameter count, the GPR should be replaced by a regressor that scales more favorably with the number of input dimensions, e.g., a deep neural network. As this also requires the presence of more training samples during the regression step, nearby regions in the similarity map could be channeled back to the neural network as additional data points. Since there is no theoretical resolution limit for the image predictions, our CNN can be retrained for higher resolutions as GPU technology improves, leaving room for exciting future improvements. To further enhance the quality of the neural network outputs, we have implemented the “late fusion” model in Karpathy et al.'s two-stream architecture [KTS+14] and experienced measurable, but marginal improvements. As this area is subject to a significant volume of followup works, we expect that this direction, alongside with rapid improvements in GPU technology will lead to the possibility of predicting outputs with more high-frequency details in full HD resolution in the near future. Variable light source types, positions, and camera angles can also be learned by the neural network to enhance the quality of gallery samples by showing animations instead of stationary images.

3.8 Conclusions

We have proposed a system for mass-scale material synthesis that is able to rapidly recommend new material models after learning the user preferences from a modest number of samples. Beyond this pipeline, we also explored combinations of the three used learning algorithms, thereby opening up the possibility of real-time material visualization, exploration and fine-tuning in a 2D latent space. Furthermore, the system works with arbitrary BSDF models and is future-proof, i.e., preference learning and recommendation works with procedural textures and displacements, where the resolution and visualization quality is expected to further improve as the graphics card compute power and on-board VRAM capacities grows over time. Throughout the scoring and recommendation steps, the users are shown noise-free images in real time and the output recommendation distribution can be controlled by a simple change of a parameter. We believe this feature set offers a useful solution for rapid mass-scale material synthesis for novice and expert users alike and hope to see more exploratory works harnessing and combining the advantages of multiple learning algorithms in the future.

Several important challenges still remain: artists with general image-processing knowledge are unable to leverage their knowledge beyond assigning scores in the initial gallery, and the rendering process for subsurface light transport remains in the order of minutes to hours, which is prohibitively long for real-time material modeling tasks. We address these challenges in the next two chapters.

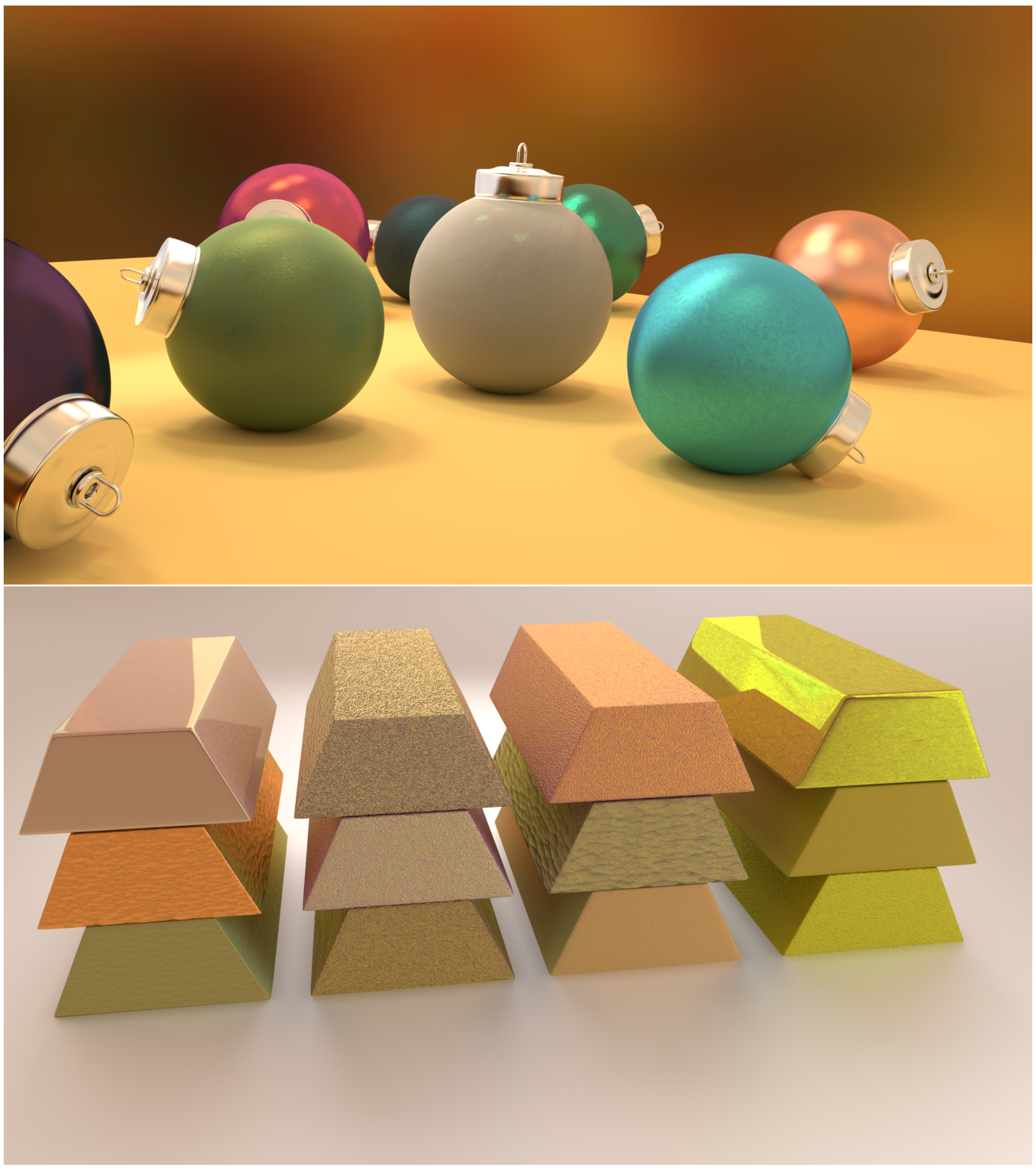


Figure 3.10: Synthesized glittery materials (above) followed by metals and minerals (below) using our extended shader.

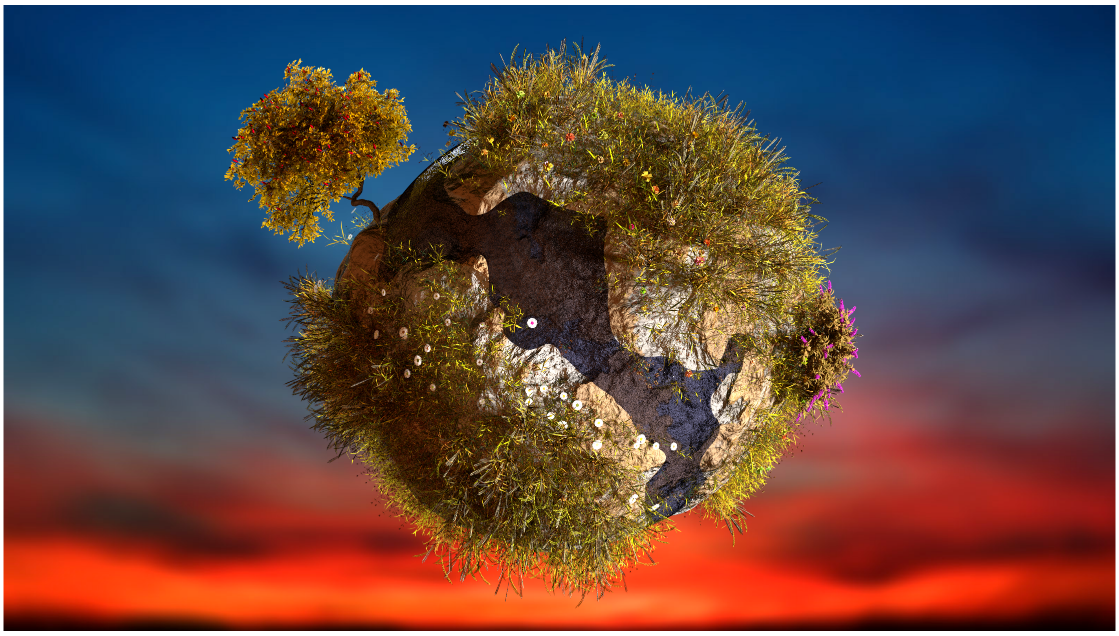
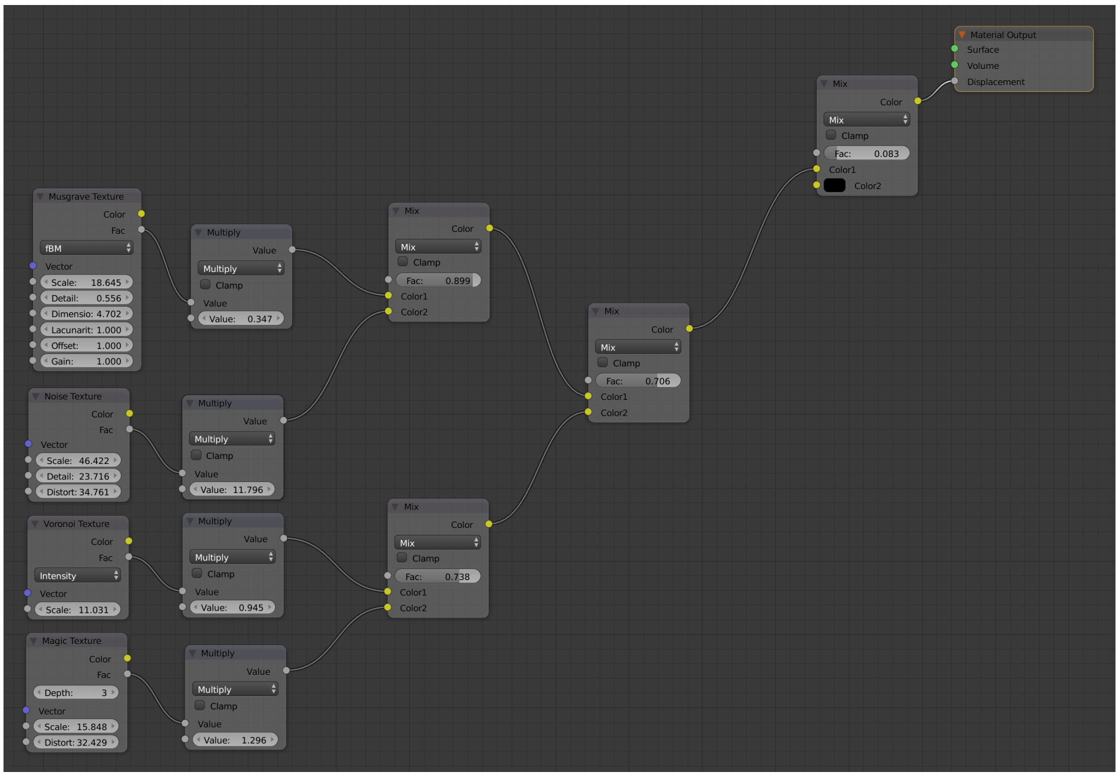


Figure 3.11: The *Microplanet* scene from the teaser image, magnified.


Musgrave Texture: fBM; Scale: 18.645; Detail: 0.556; Dimensio: 4.702; Lacunarit: 1.000; Offset: 1.000; Gain: 1.000

Multiply: Multiply; Clamp ☐; Value: 0.347

Noise Texture: Scale: 46.422; Detail: 23.716; Distort: 34.761

Multiply: Multiply; Clamp ☐; Value: 11.796

Voronoi Texture: Intensity; Scale: 11.031

Multiply: Multiply; Clamp ☐; Value: 0.945

Magic Texture: Depth: 3; Scale: 15.848; Distort: 32.429

Multiply: Multiply; Clamp ☐; Value: 1.296

Mix: Mix; Clamp ☐; Fac: 0.899

Mix: Mix; Clamp ☐; Fac: 0.738

Mix: Mix; Clamp ☐; Fac: 0.706

Mix: Mix; Clamp ☐; Fac: 0.083; Color2 ■

Figure 3.12: Our principled shader for generating a wide variety of possible displacements.

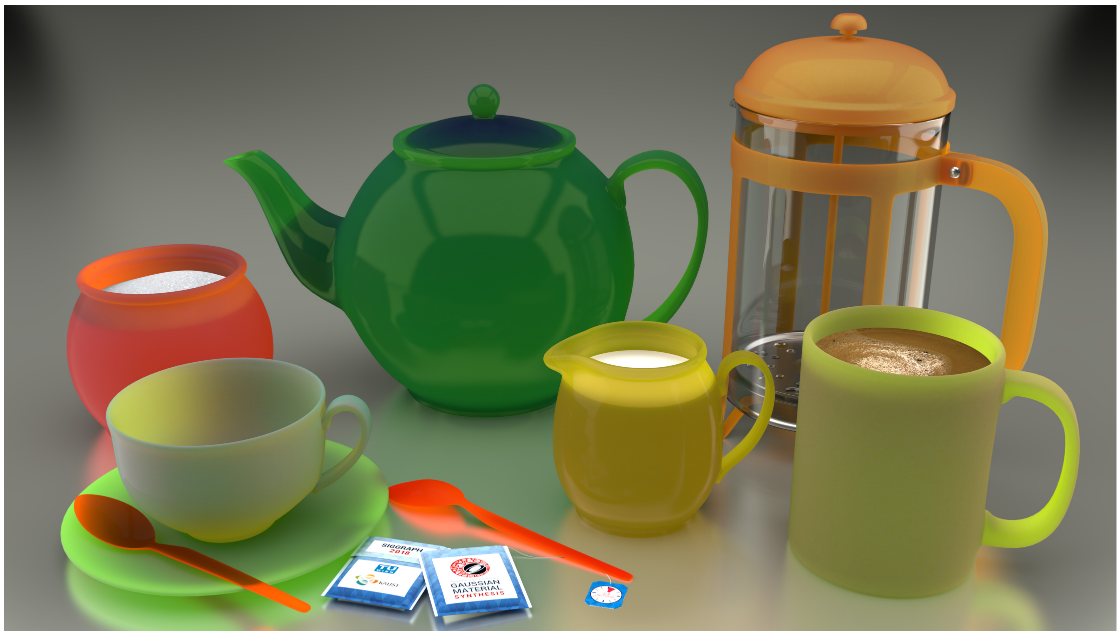


Figure 3.13: The *Toy Tea Set* scene showcasing translucent material models learned by our technique.

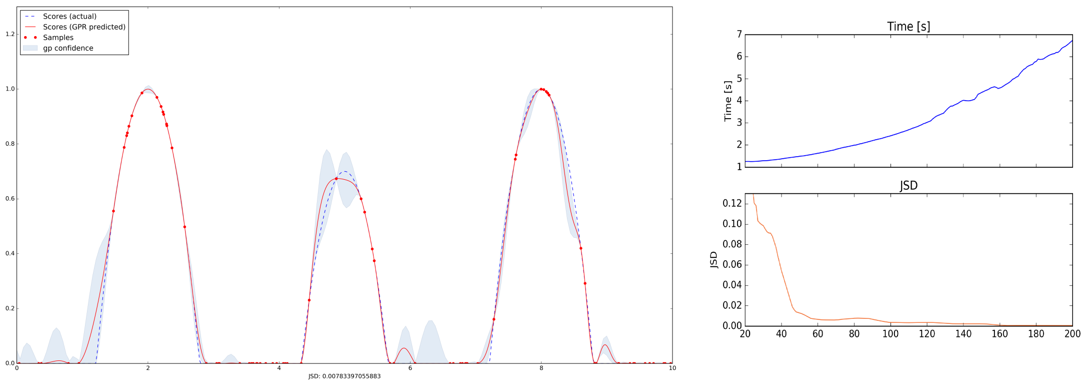


Figure 3.14: Gaussian Process Regression in 1D and the corresponding JSD and execution timings.

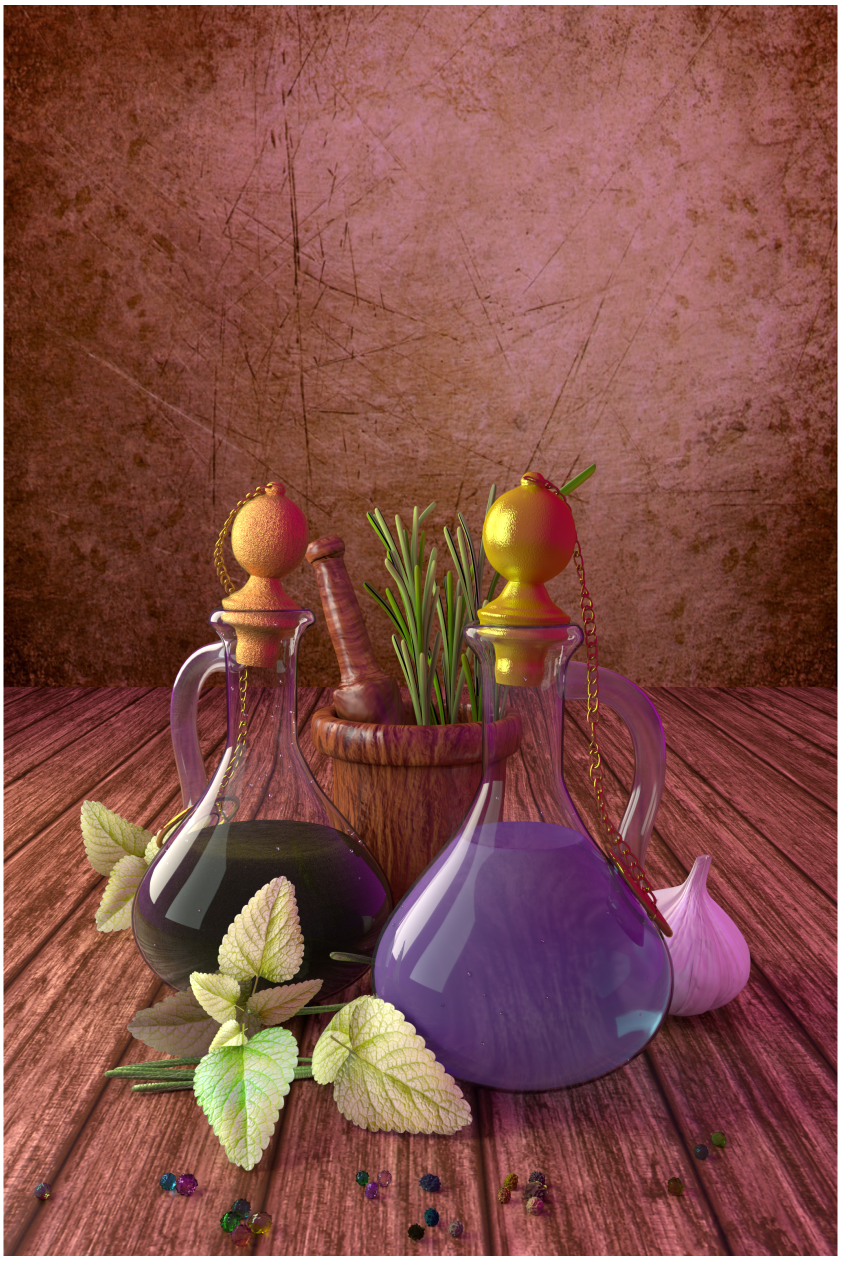


Figure 3.15: The *Still Life* scene from the teaser image, magnified.

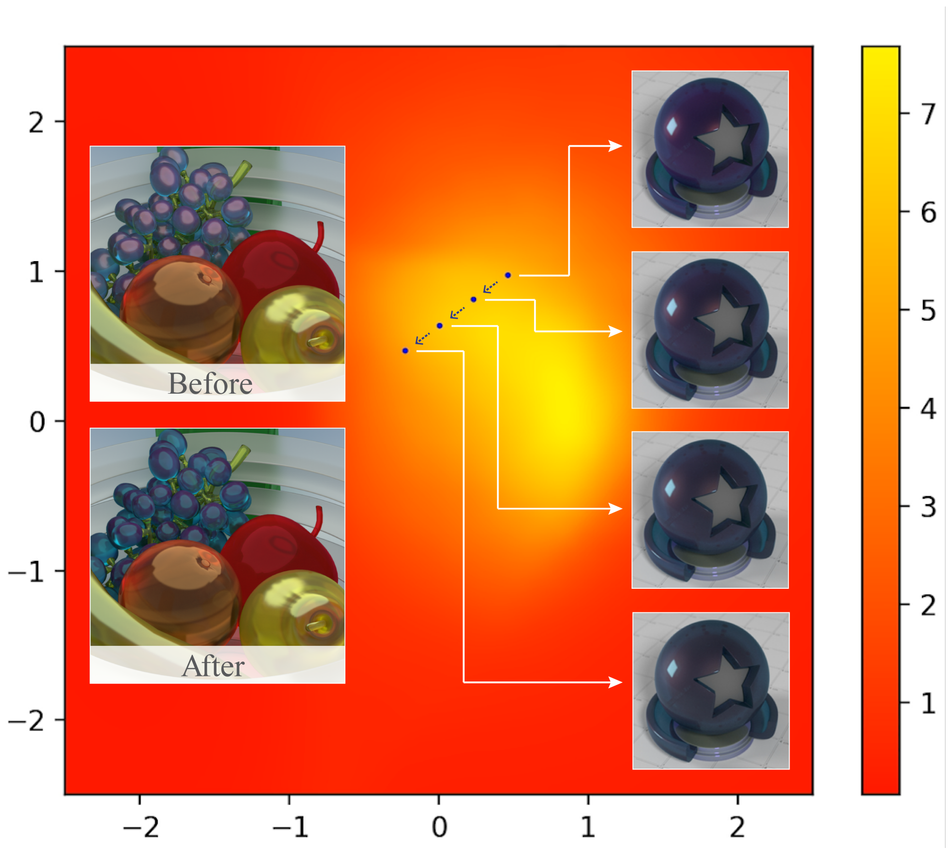


Figure 3.16: The color-coded 2D latent space can be explored in real time by the user for variant generation. The vividness of the recommended grape material can be fine-tuned rapidly without any domain knowledge.

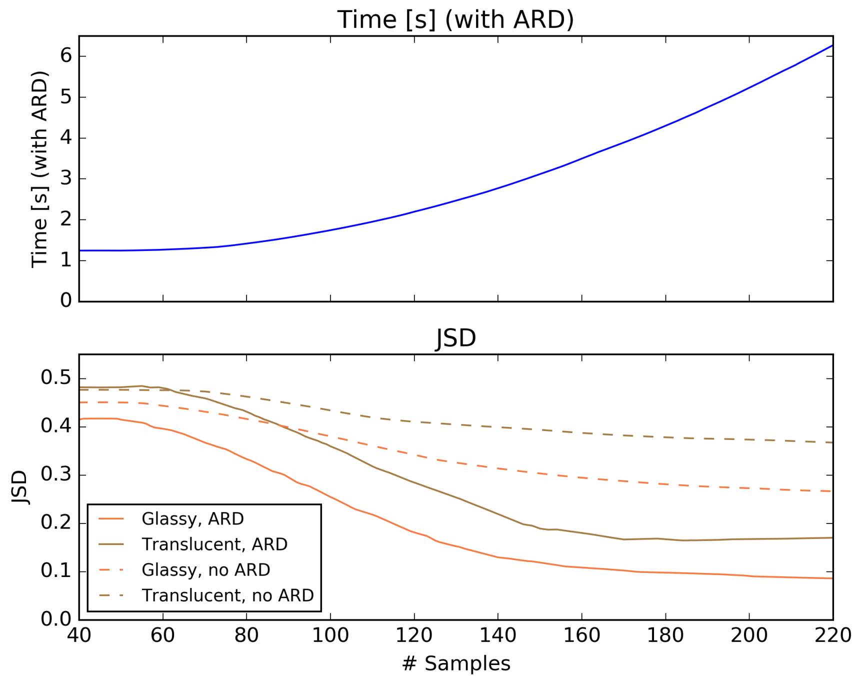


Figure 3.17: Even for more challenging cases, the presence of Automatic Relevance Determination stabilizes the GPR reconstruction quality around 150-250 training samples.

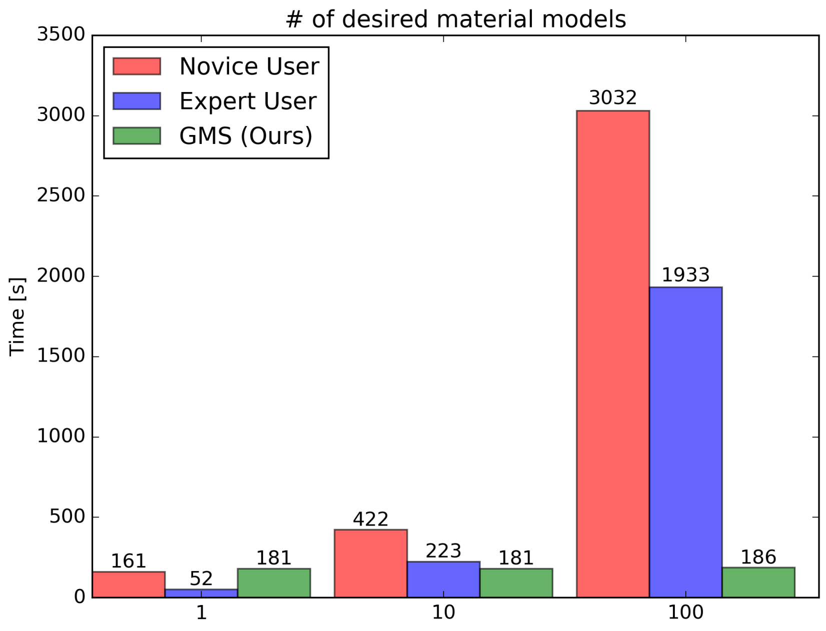


Figure 3.18: Time taken to generate 1, 10, and a 100 similar materials by hand for users of different experience levels versus our technique (with the GPR and recommendations steps).

CHAPTER 4

Photorealistic Material Editing

4.1 Motivation

In our previous work above, we have proposed one solution for the difficulties discussed in Sections 1.3.2 and 1.3.4, i.e., sped up the material visualization process by using a neural renderer and enabled novice and expert users alike to access the expressivity of principled shaders in an efficient manner. In the following work, we propose a technique that does not require the artist to assign a set of scores, and provides a solution almost immediately upon entering a mockup image of the sought material[1]. 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.

4.2 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

[1]This chapter is based on our Photorealistic Material Editing Through Direct Image Manipulation paper [ZFWW19].

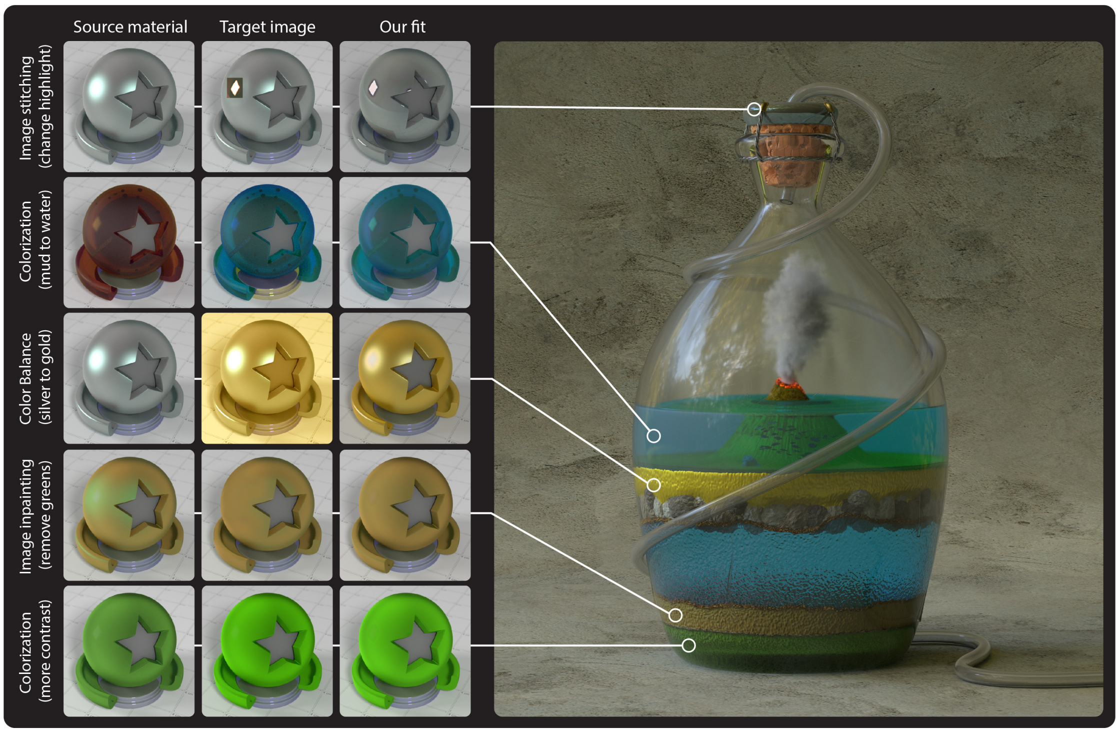

Source material
Target image
Our fit
Image stitching (change highlight)
Colorization (mud to water)
Color Balance (silver to gold)
Image inpainting (remove greens)
Colorization (more contrast)

Figure 4.1: 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. 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.

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 a reference material 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

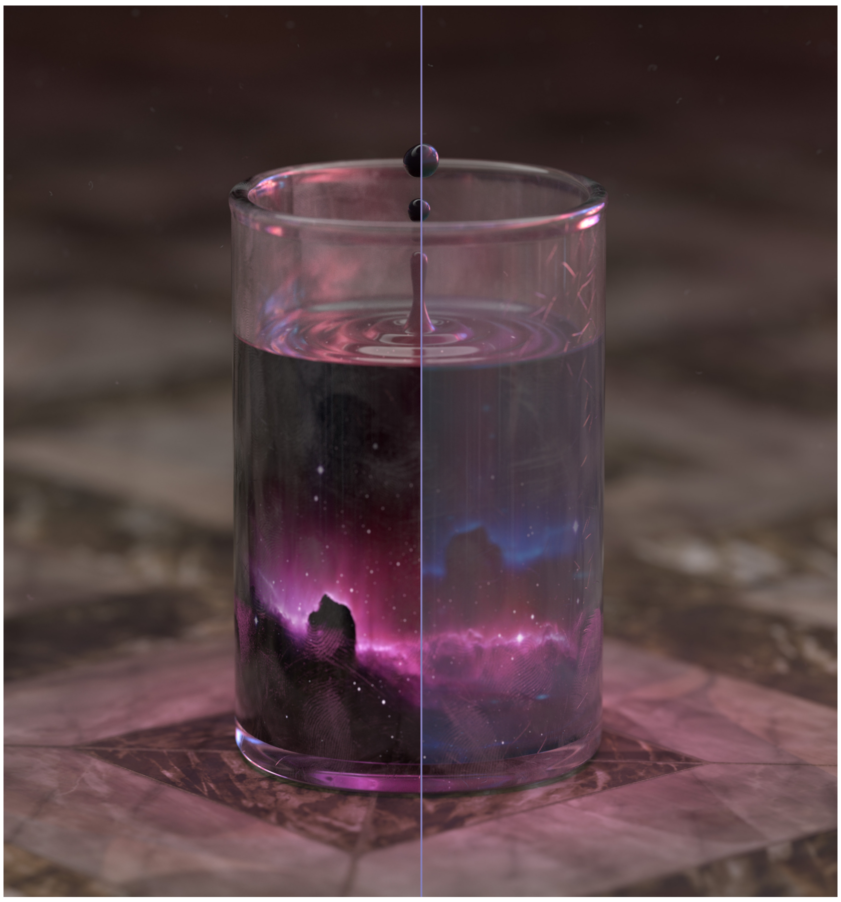


Figure 4.2: To demonstrate the utility of our system, we synthesized a new material and deployed it into an already existing scene using Blender and Cycles. In this scene, we made a material mixture to achieve a richer and foggier nebula effect inside the glass. Left: theirs, right: 50% theirs, 50% ours. Scene: Reynante Martinez.

material model can be found that is remarkably close to it (Fig. 4.3). These material models can then be easily inserted into already existing scenes by the user (Fig. 4.2).

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.

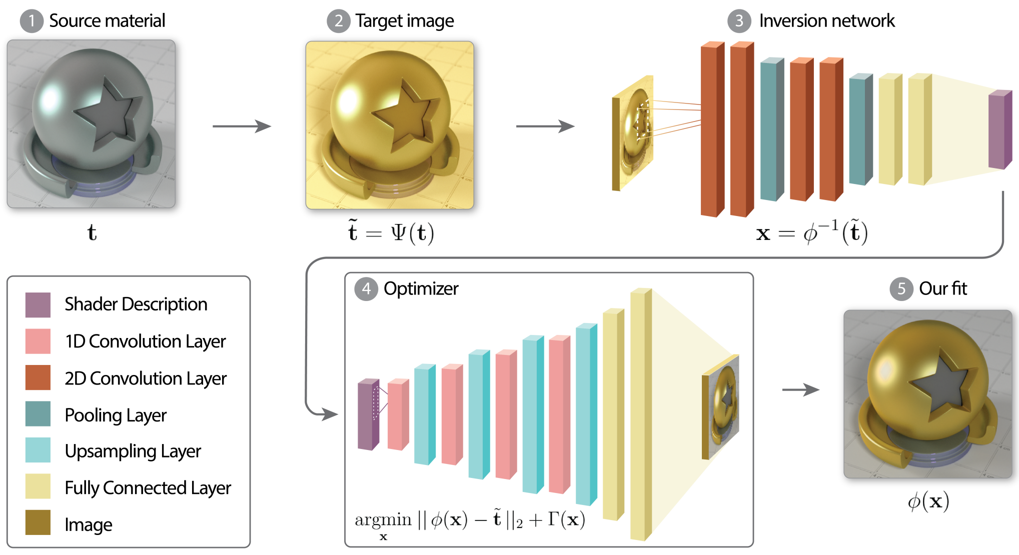


Figure 4.3: Our proposed hybrid technique offers an intuitive workflow where the artist takes a source material (❶) and produces the target image by applying the desired edits to it within a 2D raster image editor of their choice (❷). Then, one or more encoder neural networks are used to propose a set 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.

4.3 Overview

Many trained artists are adept at creating new photorealistic materials by engaging in a 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.

In this work, we set up a material test scene that contains a known lighting and geometry setup, and a fixed principled shader with a vector input of $x \in \mathbb{R}^m$ where $m = 19$.

This shader 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., $\phi\colon \mathbb{R}^m \to \mathbb{R}^p$, 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 $\tilde{\mathbf{t}}$. 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

$$\begin{aligned} \underset{\mathbf{x}}{\text{argmin}} \quad & \|\, \phi(\mathbf{x}) - \tilde{\mathbf{t}} \,\|_2, \\ \text{subject to} \quad & \mathbf{x}_{\min} \le \mathbf{x} \le \mathbf{x}_{\max}, \end{aligned} \tag{4.1}$$

where the constraints stipulate that each shader parameter has to reside within the appropriate boundaries (i.e., $0 \le x_i \le 1$ for albedos or $x_j \ge 1$ for indices of refraction). 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.,

$$\begin{aligned} & \underset{\mathbf{x}}{\text{argmin}} \ \Big(\|\, \phi(\mathbf{x}) - \tilde{\mathbf{t}} \,\|_2 + \Gamma(\mathbf{x})\Big), \text{ where} \\ & \Gamma(\mathbf{x}) = \begin{cases} 0, & \text{if } \mathbf{x} \in \mathcal{C}, \\ +\infty, & \text{otherwise,} \end{cases} \\ & \quad \mathcal{C} \ = \Big\{\mathbf{x} \mid f_i(\mathbf{x}) \ge \mathbf{0},\ i = 1, 2\Big\}, \\ & f_1(\mathbf{x}) = \mathbf{x}_{\max} - \mathbf{x}, \\ & f_2(\mathbf{x}) = \mathbf{x} - \mathbf{x}_{\min}. \end{aligned} \tag{4.2}$$

where $\mathcal{C}$ denotes the feasible region chosen by a set of constraints described by $f_i(\cdot)$ (equivalent to the second line in (4.1)) and the vector comparison operator ($\ge$) here is considered true only when all of the vector elements exceed (or equal to) zero. In a practical implementation, the infinity can be substituted by a sufficiently large integer. This formulation enabled us to compare several optimizers (Table 4.3), 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.

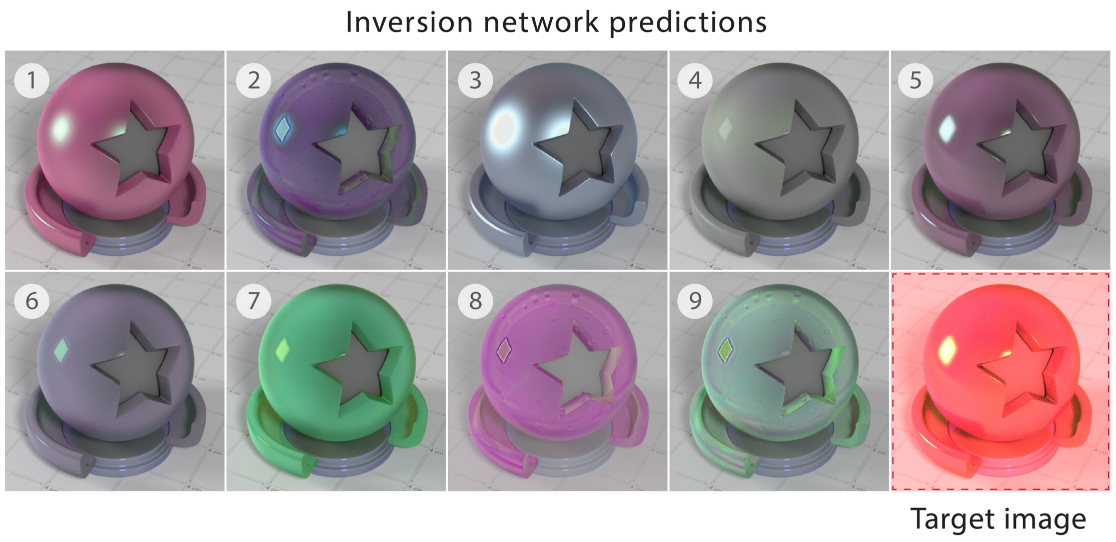


Figure 4.4: Whenever the target image strays too far away from the images contained within their training set (lower right), our 9 inversion networks typically fail to provide an adequate solution and potentially predict results outside the feasible region (❷, ❽, ❾). However, using our "best of n" scheme 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.

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 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 (4.1-4.2). 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 5.11. 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.,

$$\mathbf{x} = \phi^{-1}_{(i)}(\tilde{\mathbf{t}}), \text{ where } \quad i = \underset{j}{\operatorname{argmin}} \; ||\phi(\phi^{-1}_{(j)}(\tilde{\mathbf{t}})) - \tilde{\mathbf{t}}\,||_2, \tag{4.3}$$

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. 4.4). In Section 4.3.1 we report the structure of the neural networks used in this figure. We note that the training set for the neural renderer is equivalent to the one used in Gaussian Material Synthesis (Chapter 3), while our inversion networks are formulated as the adjoint of this neural renderer, and hence, one of their key advantage is that they can be trained on the same dataset by swapping the inputs and outputs and applying the appropriate architectural changes discussed in the manuscript.

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 (4.2) where $\mathbf{x}_{\text{init}} = \phi^{-1}(\tilde{\mathbf{t}})$ and a more detailed description of this hybrid solution is given below in Algorithm 4. 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.

As the first stage executes within a few milliseconds, it can be used as-is for real-time applications where an approximate solution is tolerable. 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 both 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.

Algorithm 4 Photorealistic material editing

```
 1: Given $\mathbf{t}$, $\phi(\cdot)$, $[\phi^{-1}_{(1)}(\cdot), \dots, \phi^{-1}_{(n)}(\cdot)]$, $\mathbf{x}_{\text{min}}$, $\mathbf{x}_{\text{max}}$
 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 $f_1(\mathbf{x}) = \mathbf{x}_{\text{max}} - \mathbf{x}$                         ▷ Set up constraints
 8: Define $f_2(\mathbf{x}) = \mathbf{x} - \mathbf{x}_{\text{min}}$
 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
```

Predicting image sequences. A typical image editing workflow 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 4 to add

$$\phi^{-1}_{(n+1)}(\tilde{\mathbf{t}}_k) = \underset{\mathbf{x}}{\operatorname{argmin}} \Big(||\phi(\mathbf{x}) - \tilde{\mathbf{t}}_{k-1}||_2 + \Gamma(\mathbf{x}) \Big). \tag{4.4}$$

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 5.11, 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.

4.3.1 Neural network architectures

Below, we describe the neural network architectures we used to implement each individual $\phi^{-1}_{(i)}$ shown in Fig. 4.4. 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].

	Initial guess		50 fun. evals		300 fun. evals		1500 fun. evals	
Input	Random	NN	Optimizer	Ours	Optimizer	Ours	Optimizer	Ours
Fig. 4.5, Row 1	41.93	5.94	33.81	**4.53**	9.42	**2.84**	5.62	**2.37**
Fig. 4.5, Row 2	78.45	32.72	68.55	**32.67**	40.24	**32.67**	40.21	**32.67**
Fig. 4.5, Row 4	35.37	18.68	30.88	**16.53**	17.29	**14.71**	16.98	**14.68**
Fig. 4.5, Row 7	41.65	22.42	38.10	**22.38**	26.30	**22.38**	26.24	**22.38**
Fig. 4.5, Row 8	29.04	19.82	26.79	**18.43**	22.93	**15.37**	22.93	**15.37**
Fig. 4.8, Row 2	23.78	12.79	20.31	**11.62**	8.27	**7.81**	8.26	**7.80**
Fig. 4.8, Row 3	21.60	9.09	16.54	**8.28**	6.24	**5.80**	6.19	**5.80**
Fig. 4.8, Row 8	29.58	9.74	22.69	**7.92**	6.63	**5.36**	6.63	**5.36**

Table 4.1: 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. 4.5 and 4.8. The numbers indicate the RMSE of the outputs, and for reference, the first row showcases an input image that is reproducible by the shader.

4.4 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

		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 4.2: Our proposed reinitialization technique consistently outperforms per-frame computation for the image sequence shown in Fig. 4.6. The numbers indicate the RMSE of the outputs.

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 (4.2) 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., $\mathbf{x} \approx \phi^{-1}_{(i)}(\tilde{\mathbf{t}})$ as shown in (4.3),
- our **hybrid method** is obtained by combining the two above approaches as described in Algorithm 4.

At the end of this section, we also compare the total time taken to synthesize 1, 10, and 100 selected materials against Gaussian Material Synthesis (Chapter 3).

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 (Fig. 4.4). Our goal with this experiment was to demonstrate that a solution containing only one inversion network generally produces unsatisfactory outputs, regardless of network structure. However, 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.

Optimizer and hybrid solution accuracy. In Table 4.1, 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. 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 (Table 4.3), 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. 4.5, 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.

Supported 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. We cluster the set of possible transforms into *global* (Fig. 4.5) and *local* (Fig. 4.8) operations: these cases include saturation increase, grayscale transform, colorization, image mixing, stitching and inpainting, and selective blurring of highlights. 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. Throughout this work, for each comparison, the RMSE is reported, which is widely regarded as the standard way of measuring differences in BRDF modeling [MPBM03, DJ18]. There are specialized cases, e.g., noise and blurring among other examples, that require non-standard image-quality metrics [ZM09, LWC$^+$13] – regardless, we have tried measuring the PSNR and produced per-channel greyscale images to record the SSIM [WBS$^+$04] and have not found meaningful differences to RMSE in our test cases.

Image sequence prediction. As our earlier results in Table 4.1 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. 4.6). 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 4.2, 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. 4.7, we have recorded the modeling times for 1, 10, and a 100 similar materials using our method and compared them against Gaussian Material Synthesis (GMS, Chapter 3), a learning-based technique for mass-scale material synthesis. We briefly describe the most important parameters of the recorded execution times and refer the interested reader to this paper for more details – the novice and expert user timings were taken from the GMS paper and indicate the amount of time these users took to create the prescribed number of materials by hand using Disney's "principled" shader [BS12], whereas GMS and our timings contain both the modeling (i.e., scoring a material gallery in GMS and performing image processing for our technique) and 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 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. In the case of mass-scale material synthesis, i.e., 100 or more materials, both methods outperform experts, where GMS offers the best scaling solution. In each case, the timings for our technique include the fixed cost of loading the 9 neural networks (5.5s). Throughout this manuscript, all results were generated using a NVIDIA TITAN RTX GPU.

Comparison of optimizers. In Table 4.3, 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.

4.5 Limitations and Future Work

As demonstrated in Fig. 4.4, the results of ϕ^{-1} depend greatly on the performance of the encoder and decoder neural networks. As these methods enjoy significant research

Input	Init. type	Init. RMSE	Nelder-Mead	L-BFGS-B	SLSQP	CG	Basin-hopping
Fig. 4.5, Row 1	Rand	41.93	5.62	20.47	17.96	5.24	**2.01**
Fig. 4.5, Row 1	NN	5.94	2.37	5.84	5.94	5.94	
Fig. 4.5, Row 2	Rand	78.45	40.21	78.45	78.45	78.45	**32.67**
Fig. 4.5, Row 2	NN	32.72	**32.67**	32.72	32.72	32.72	
Fig. 4.5, Row 4	Rand	35.37	16.98	28.84	35.37	34.99	14.72
Fig. 4.5, Row 4	NN	18.68	**14.68**	15.33	18.18	15.90	
Fig. 4.5, Row 7	Rand	41.65	26.24	41.65	41.65	41.65	**22.38**
Fig. 4.5, Row 7	NN	22.42	**22.38**	22.42	22.42	22.42	
Fig. 4.5, Row 8	Rand	29.04	22.93	29.04	26.71	28.21	15.69
Fig. 4.5, Row 8	NN	19.82	**15.37**	19.82	28.87	19.82	
Fig. 4.8, Row 2	Rand	23.78	8.26	23.78	23.78	21.75	**7.63**
Fig. 4.8, Row 2	NN	12.79	7.80	12.79	12.79	12.79	
Fig. 4.8, Row 3	Rand	21.60	6.19	21.60	21.60	20.83	5.86
Fig. 4.8, Row 3	NN	9.09	**5.80**	9.09	9.09	9.09	
Fig. 4.8, Row 8	Rand	29.58	6.63	29.58	29.58	29.58	**5.07**
Fig. 4.8, Row 8	NN	9.74	5.36	9.61	9.61	9.68	

Table 4.3: 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.

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]. 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 $\tilde{\mathbf{t}}$.

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 a classic 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], and more.

4.6 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. 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 work. 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 4.9). Our two 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. 4.3). This combination results in a two-stage sytem that opens up efficient material editing workflows for artists without expertise in this area.

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.

For users that seek more than a handful materials and wish to engage in rapid variant generation, we recommend using **Gaussian Material Synthesis**, while we designed this work, i.e., **Photorealistic Material Editing Through Direct Image Manipulation** for artists who seek only a few materials and wish to leverage their knowledge in 2D raster image editing to save more time. We would like to thank Reynante Martinez for providing us the geometry and some of the materials for the Paradigm (Fig. 4.1) and

Genesis scenes (Fig. 4.2), ianofshields for the Liquify scene that served as a basis for Fig. 4.9, 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 NVIDIA for providing the GPU used to train our neural networks. This work was partially funded by Austrian Science Fund (FWF), project number P27974.

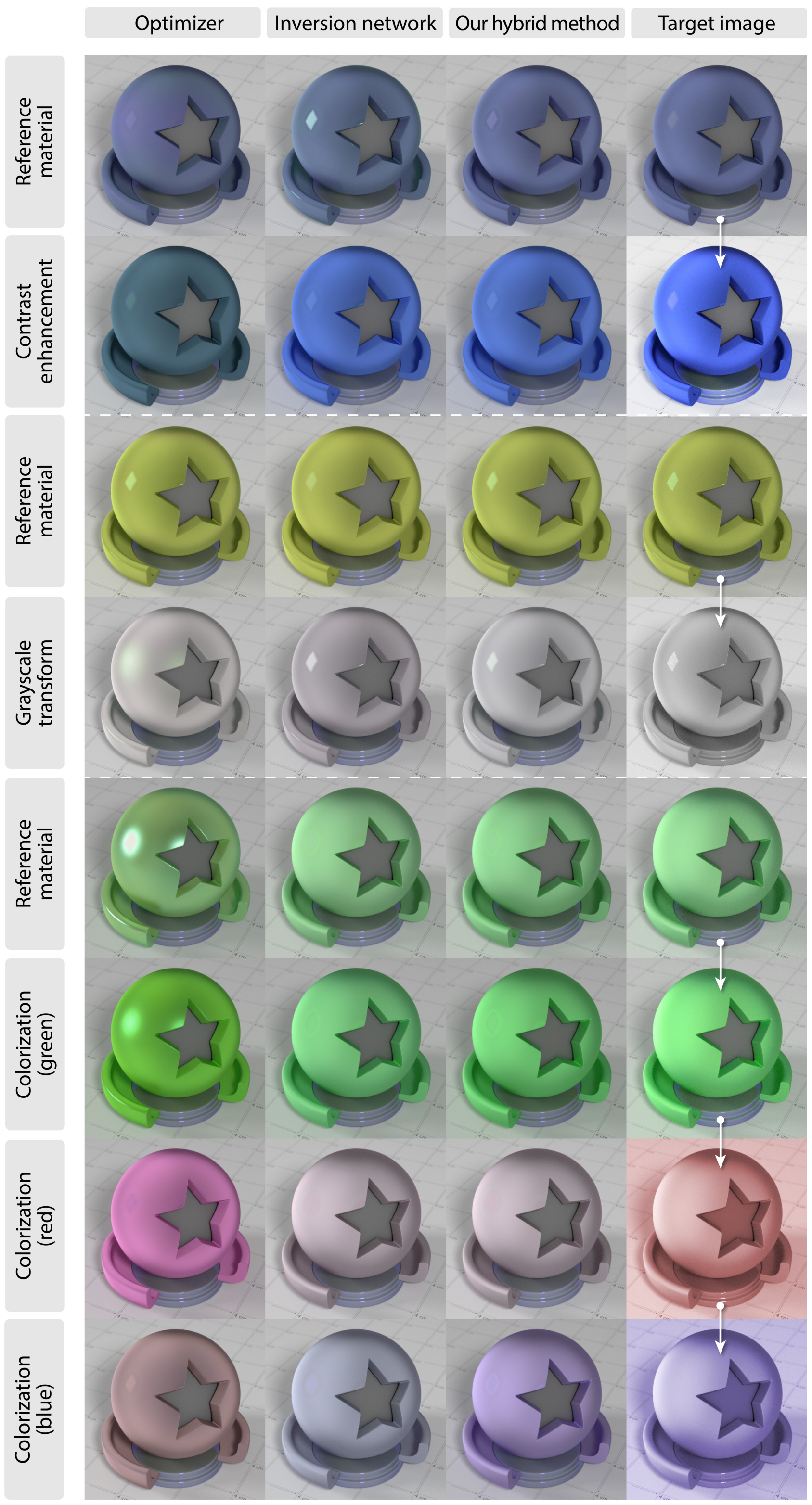


Figure 4.5: Results for three techniques on common global colorization operations including saturation increase and grayscale transform. The “reference material” labels showcase materials that can be obtained using our shader and are used as source images for the materials below them, where the arrows denote the evolution of the target image.

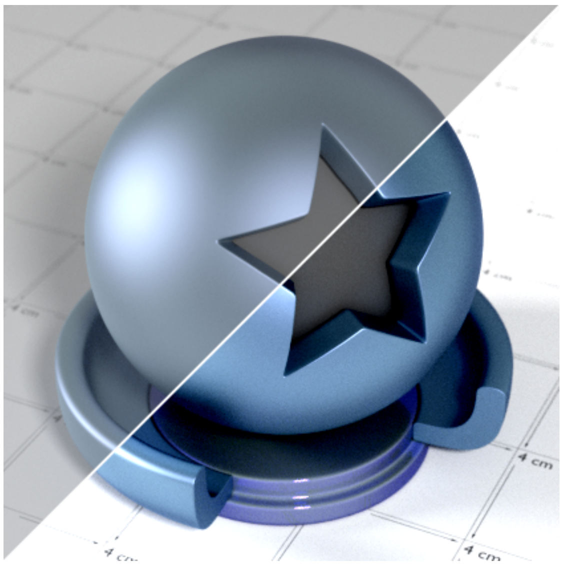


Figure 4.6: Our image sequence starts with an input that is achievable using our shader (upper left), where each animation frame slightly increases its black levels. The lower right region showcases the 300th frame of the animation.

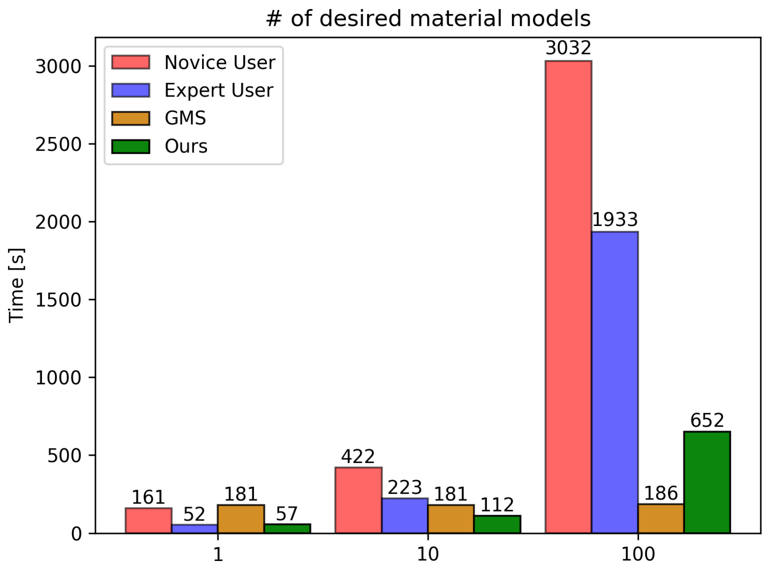


Figure 4.7: The recorded modeling times reveal that if at most a handful (i.e., 1-10) of target materials are sought, our technique offers a favorable entry point for novice users into the world of photorealistic material synthesis.

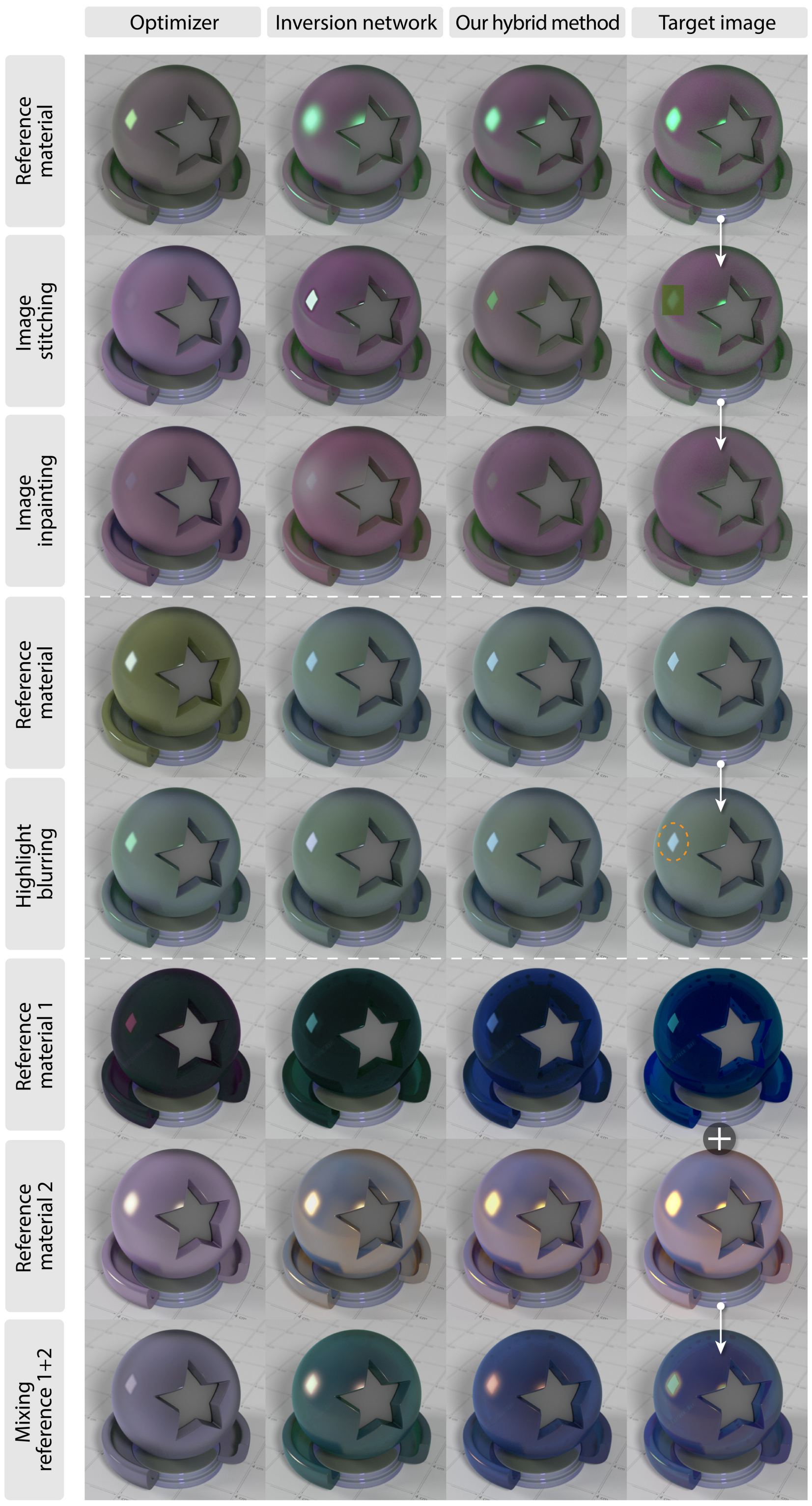


Figure 4.8: Results for three techniques on local image editing operations and image mixing. The "reference material" labels showcase materials that can be obtained using our shader and are used as source images for the materials below them, where the arrows denote the evolution of the target image.

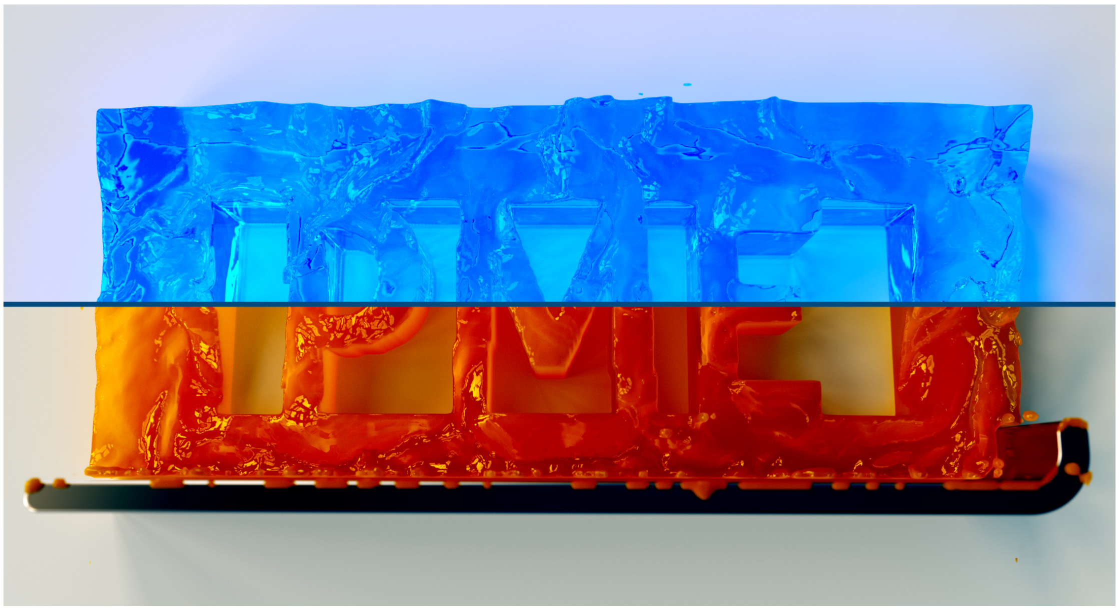


Figure 4.9: Our technique is especially helpful early in the material design process where the user seeks to rapidly iterate over a variety of possible artistic effects. Both material types were synthesized using our described method. We demonstrate this workflow in our supplementary video.

CHAPTER 5

Modeling Real-Time Subsurface Scattering

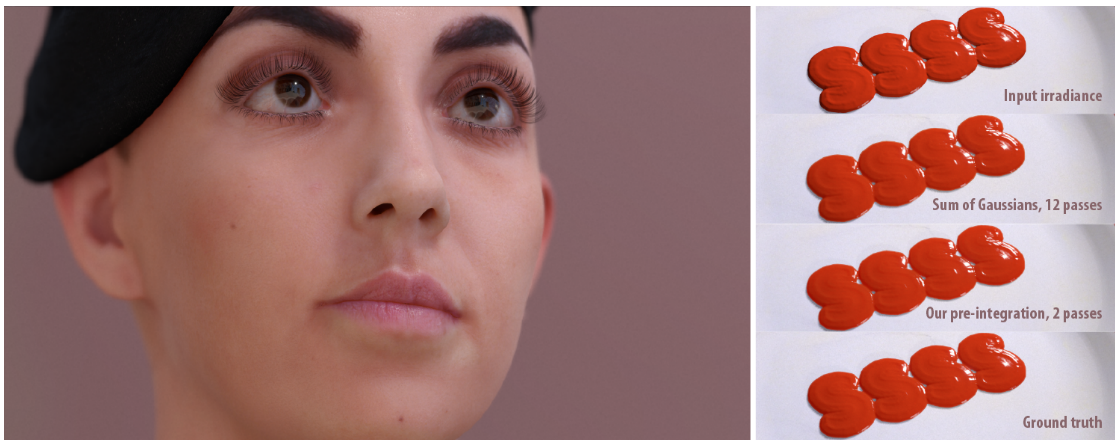


Figure 5.1: Real-time results of our method for simulating translucent materials (skin on the left, ketchup on the right). Our separable subsurface-scattering method enables the generation of these images using only two convolutions (versus 12 in the sum-of-Gaussians approach [dLE07, JSG09]) and seven samples per pixel, while featuring quality comparable with the current state of the art, at a fraction of its cost. It can be implemented as a post-processing step and takes only 0.489 ms per frame on an AMD Radeon HD 7970 at 1080p, which makes it highly suitable for challenging real-time scenarios.

5.1 Motivation

So far, we have shown two works on material synthesis and editing, but haven't addressed the issue of rendering subsurface light transport. In 1.3.6, we discussed why rendering one accurate image of a translucent material often takes from minutes to hours. To address this problem, in this work, we propose two real-time models for simulating subsurface scattering for a large variety of translucent materials, which need under 0.5 milliseconds per frame to execute. This makes them a practical option for real-time production scenarios[1]. Current state-of-the-art, real-time approaches simulate subsurface light transport by approximating the radially symmetric non-separable diffusion kernel with a sum of separable Gaussians, which requires multiple (up to twelve) 1D convolutions. In this work we relax the requirement of radial symmetry to approximate a 2D diffuse reflectance profile by a single separable kernel.

We first show that low-rank approximations based on matrix factorization outperform previous approaches, but they still need several passes to get good results. To solve this, we present two different separable models: the first one yields a high-quality diffusion simulation, while the second one offers an attractive trade-off between physical accuracy and artistic control. Both allow rendering subsurface scattering using only two 1D convolutions, reducing both execution time and memory consumption, while delivering results comparable to techniques with higher cost. Using our importance-sampling and jittering strategies, only seven samples per pixel are required. Our methods can be implemented as simple post-processing steps without intrusive changes to existing rendering pipelines.

5.2 Introduction

The accurate depiction of translucent materials is an important but challenging topic in the motion picture and video game industries. Rendering realistic subsurface scattering (SSS) implies simulating how light travels and scatters inside translucent media, which is an expensive process. While offline rendering scenarios can afford longer computation times, real-time applications, such as video games, impose severe time constraints, often leading to the exclusion of subsurface scattering and translucency effects. This in turn hinders the level of realism that can be achieved.

One of the most common approaches to compute subsurface scattering efficiently exploits the fact that it blurs high-frequency details and illumination. This means that simulating subsurface scattering can be approximated as a convolution with a diffusion kernel that mimics the diffuse reflectance profile for a given translucent medium. While the exact reconstruction normally requires an expensive two-dimensional convolution, d'Eon et al. [dLE07] showed that it can be approximated by a sum of radially symmetric Gaussians. Thus, due to the separability of Gaussians, the 2D convolution can be computed using a set of cheaper 1D passes, which allows high-quality skin rendering in real time. This approach was later extended to screen space, modulating the width of the kernel according to per-pixel depth information [JSG09].

[1]This chapter is based on our equivalently named paper [JJG15].

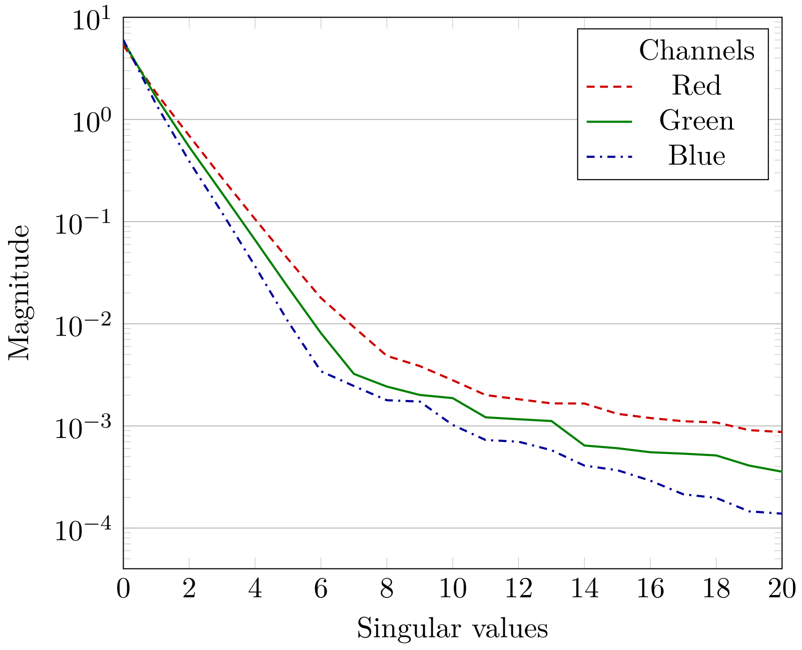


Figure 5.2: Decay of the singular values in the singular value decomposition of a diffuse reflectance profile used to simulate subsurface scattering in skin. Only the components associated to the first few singular values contribute appreciably to the reconstruction of the profile, making a low-rank approximation feasible.

However, in order to get adequate results, several Gaussians are needed to model the diffuse reflectance profile. This translates into multiple 1D convolutions per frame, which is still costly. In this work, we make the key observation that exact simulated diffusion kernels, which are in general mathematically non-separable, can be closely reconstructed by a low-rank factorization for a wide range of materials (see Figure 5.2). Based on this, we present two different separable models that allow simulating subsurface scattering with just two 1D convolutions: The first one allows reconstructing a high-quality diffusion profile based on the observation that the irradiance is close to be additively separable, while the second is an artist-friendly model that, following the previous observation, provides an attractive trade-off between physical accuracy and ease of use for artistic editing of the scattering profiles (Table 5.1). Coupled with our importance-sampling and jittering strategies, our methods only require seven samples per pixel (see Figures 5.1 and 5.43).

Our methods can be implemented as simple post-processing steps and do not rely on complex alpha-blending pipelines or Gaussian levels of detail [JG10], and work with dynamic objects without any additional cost. Moreover, all our rank-1 approximations execute in less then 0.5 ms per frame on modern commodity hardware and exhibit negligible fixed costs, as regions of the scene with no visible scattering can be quickly

Model	Visual quality	1D convolutions	Separable
1 Gaussian	low	2	✓
2+ Gaussians	high	2 per Gaussian	✗
Kernel pre-integration	high	2	✓
Artist-friendly model	controllable	2	✓

Table 5.1: Compared to the state of the art [dLE07, JSG09] (above the separation line), our proposed techniques (below) offer solutions for a variety of trade-off choices. Our techniques are able to provide high-quality results by using a separable approximation of only two 1D convolutions.

culled using stencil buffering. Our separable approximation of subsurface scattering fills the gap between physically based subsurface-scattering rendering and highly time-constrained environments such as games, and it is currently being used in game engines and production pipelines. We additionally provide the source code for the entirety of this project.

5.3 Separable Subsurface Scattering

The diffuse reflectance of a homogeneous translucent material due to subsurface light scattering is characterized by its 2D diffuse reflectance profile $R_d(x, y)$, which describes the light reflected around a normally incident pencil beam on the origin of a surface of an infinite half-space [JMLH01]. For a homogeneous material, R_d is radially symmetric, and can be characterized by a 1D diffusion profile $R_d(r)$ such that $R_d(x, y) = R_d(\|(x, y)\|)$[2]. It can be used to calculate the radiant exitance $M_e(x, y)$ at an arbitrary surface point (x, y) according to:

$$M_e(x, y) = \int_{\mathbb{R}^2} E(x', y')\, R_d(x - x', y - y')\, dx'dy', \tag{5.1}$$

where $E(x, y)$ is the irradiance at point (x, y), and both $M_e(x, y)$ and $E(x, y)$ are measured in Wm^{-2}. Note that Equation (5.1) has the form of a 2D convolution with the 2D reflectance profile: $M_e(x, y) = (E * R_d)(x, y)$.

Approximation of 2D diffusion profiles. For real-time applications, carrying out the 2D convolution in Equation (5.1) is prohibitively expensive. However, if we can express the profile R_d as an approximation A consisting of a sum of *separable* functions, it is possible to approximate this operation by a sequence of $2N$ 1D convolutions, which

[2]Note that the radial symmetry of the diffusion assumption follows from assuming a normally incident incoming light, which in general does not hold in real applications. For a description of SSS with directional-dependent diffusion we refer the reader to e.g. [DLR+09, HCJ13, FHK15].

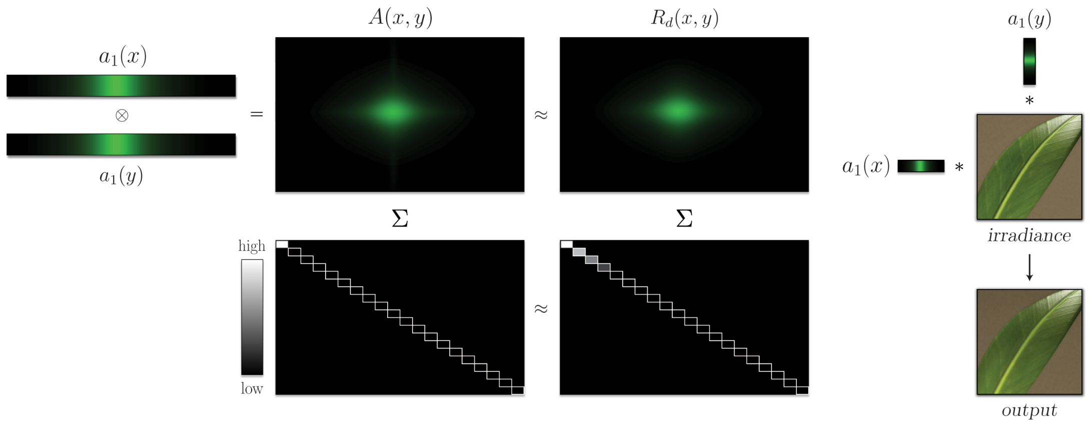


Figure 5.3: Overview of our approach: based on the low-rank nature of the diffusion kernel $R_d(x, y)$, shown by the Σ matrix below storing the singular values of the kernel (magnitude in grayscale), we approximate $R_d(x, y)$ with $A(x, y) = a_1(x)a_1(y)$. This simplifies the simulation of subsurface scattering (right) to just two 1D convolutions per summand with the irradiance signal.

exhibit significantly smaller computational complexity:

$$(E * R_d)(x,y) \approx (E * A)(x,y) = \sum_{i=1}^{N}((E * a_i) * a_i)(x,y)$$
$$\text{with } A(x,y) = \sum_{i=1}^{N} a_i(x)a_i(y), \tag{5.2}$$

where the approximation A is defined by 1D functions a_i. From the radial symmetry of R_d it follows that the same functions a_i can be employed in both coordinate directions. d'Eon et al. [dLE07] observed that zero-mean Gaussians G are suitable functions for approximation:

$$R_d(x,y) \approx A_g(x,y) = \sum_{i=1}^{N} w_i G(x,y;\sigma_i), \tag{5.3}$$

where σ_i denote the standard deviation of the respective Gaussians. Due to the separability of the Gaussian kernel, the convolution with A_g can be realized as $2N$ 1D convolutions. Unfortunately, for the most demanding real-time scenarios these $2N$ convolutions are still too expensive.

In practice, however, we only work with discretized diffusion kernels, which can be interpreted and analyzed as 2D matrices. This allows us to make the observation that most of the energy of typical diffusion kernels is stored in the first few singular values, and in particular in the first one (see Figure 5.2). This means that the diffusion profile can be approximated with a low-rank separable approximation, and that a single separable kernel can reproduce most of the kernel perceptual qualities, if chosen appropriately. The application of such a separable kernel for rendering is illustrated in Figure 5.3.

An obvious choice for a discretized separable kernel would be using just the first component of the *singular value decomposition* (SVD), which – according to the Eckhart-Young theorem [EY36] – gives the best low-rank approximation with respect to the Frobenius norm. Unfortunately, the rank-1 approximation of the kernel using SVD exhibits a rather large energy loss and produces unsatisfactory results even if energy conservation is enforced by normalizing the kernel. This is due to the fact that a pure kernel-space factorization does not take into account that in image-space, some parts of the kernel are more relevant than others. Higher-rank SVD-based approximations (i.e., $N \approx 2 - 6$) converge very rapidly to the original kernel (Figure 5.4), but the increased computation times make it a less attractive option for real-time applications. In the following we show that, under certain assumptions, even a rank-1 approximation can be used to reconstruct the diffusion kernel with high accuracy (Section 5.4), and then propose an artist-friendly separable model that allows intuitive editing of the appearance of translucent materials (Section 5.5).

5.4 Pre-integrated Separable Kernel

Both the sum-of-Gaussians approximation and our SVD-based method produce unsatisfactory results for a single summand, i.e. $N = 1$. Due to the non-separability of discretized representations of realistic diffusion profiles, it is not possible to fully reconstruct the effect of their convolution with 2D signals by a single separable kernel. Additionally, separable approximation kernels are in general not radially symmetric, as illustrated by

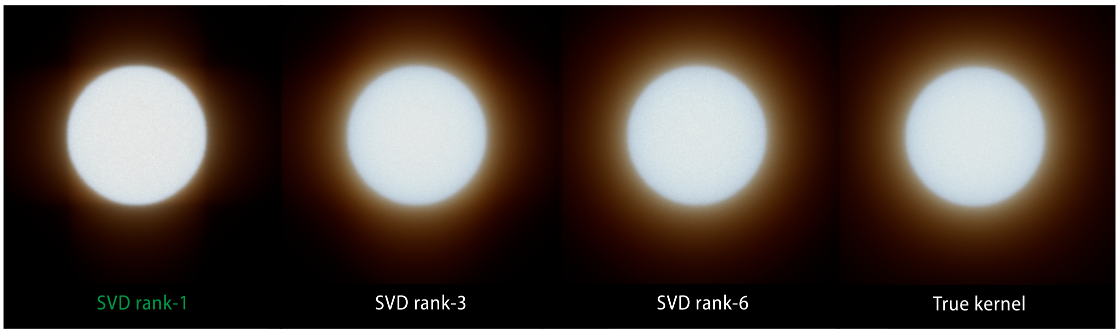


Figure 5.4: Results of rank-N approximations obtained using the SVD of the discrete diffusion profile for skin, for $N = \{1, 3, 6\}$. Using the SVD's rank-1 (separable) approximation leads to poor results, since most of the kernel's energy is stored in the center of the kernel. Increasing the rank of the approximation leads to a more faithful approximation of the diffusion kernel, but at the cost of introducing several passes, which makes it inefficient for time-constrained applications. We refer the reader to Section 5.7 for more results using the SVD low-rank approximation.

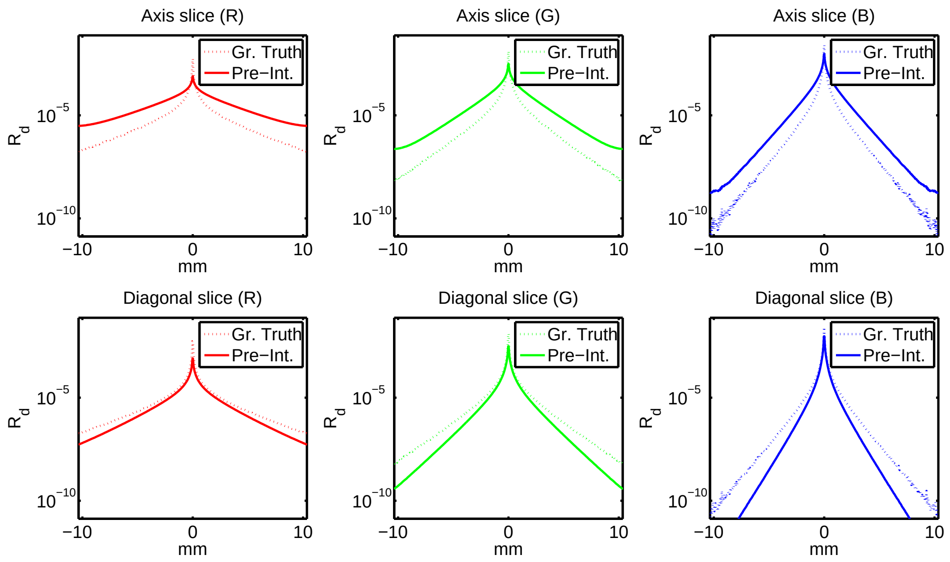


Figure 5.5: Plot of our pre-integrated kernel compared to the ground truth for human skin, in both the axial and diagonal directions; it can be seen that due to the loss of radial symmetry, our kernel gives different results for the axial and diagonal directions, as opposed to the ground-truth kernel. Note that our method does not try to mimic the kernel to be close to the 2D diffusion kernel, but it tries to match the final result of the convolution (Figure 5.6 and 5.7). Additional comparisons can be found in the Section 5.7.

an example in Figure 5.5. It is, however, possible to completely reproduce a profile's behavior on a special class of signals: assuming that the irradiance is additively separable, i.e., $E(x, y) = E_1(x) + E_2(y)$ or, equivalently, $\frac{\partial E}{\partial x \partial y} = \frac{\partial E}{\partial y \partial x} = 0$, the radiant exitance M_e is given as follows (refer to Section 5.7. for details):

$$\begin{aligned} M_e(x, y) &= \iint E(x', y')\, R_d(x - x', y - y')\, dx'dy' \\ &= \int E_1(x') \underbrace{\int R_d(x - x', y - y')\, dy'}_{a_p(x-x')} dx' \\ &\quad + \int E_2(y') \underbrace{\int R_d(x - x', y - y')\, dx'}_{a_p(y-y')} dy' \\ &= \iint E(x', y') \frac{1}{\|a_p\|_1} a_p(x - x') a_p(y - y')\, dx'dy', \end{aligned} \tag{5.4}$$

where a_p denotes the pre-integrated 1D kernel of R_d along a coordinate axis. Due to the radial symmetry of R_d, we have $a_p(x) = a_p(y)$, where $\|a_p\|_1 = \|R_d\|_1$ by definition.

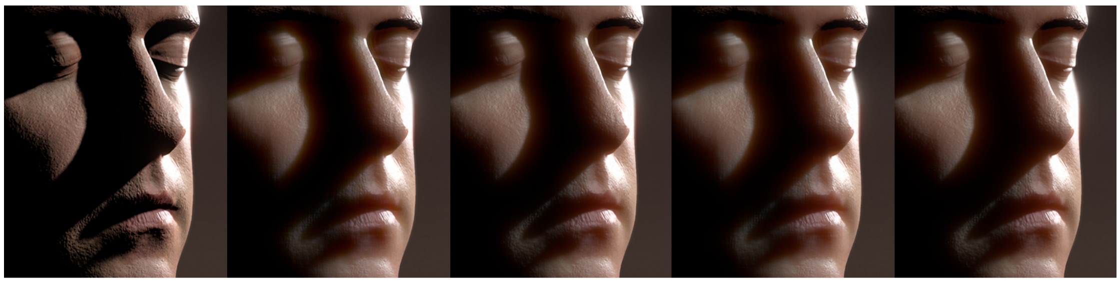

Input
Artist-friendly model (close fit)
Sum of Gaussians (6 Gaussians)
Kernel pre-integration
Ground truth

Figure 5.6: Comparison of the different techniques proposed with a six-Gaussian fit for skin [dLE07, JSG09] and the ground truth (2D kernel). Note that our separable approximations lead to similar quality results with just two 1D convolutions, as opposed to the twelve needed by the sum-of-Gaussians approach.

Hence, we define the pre-integrated[3]kernel A_p of the diffusion profile as:

$$A_p(x, y) = \frac{1}{\|R_d\|_1} a_p(x) a_p(y). \tag{5.5}$$

Note that A_p reproduces the *exact* 2D convolution with R_d in the presence of additively separable irradiance signals, such as straight shadow boundaries of arbitrary orientation or general axis-aligned 1D functions (see Figure 5.7 for an example of a vertical shadow boundary).

Even for general (i.e., non-additively separable) signals, this approximation yields good results for a wide range of materials and scenarios (see Figure 5.6 and 5.43), since most real-world signals $E(x, y)$ can be locally approximated with additively separable functions. However, this formulation offers limited control to an artist and needs to be discretized to be used in practical applications; in the following section we describe a separable, *artist-friendly* model that overcomes these two limitations.

5.5 An Artist-Friendly Separable Model

Our pre-integrated kernel (Section 5.4) is able to reconstruct a wide range of materials given its diffusion kernel, which can be obtained from measured data [JMLH01, MES+11] or from simulations. However, in a production environment these profiles might not be the optimal ones, since they might not match the assets (e.g., color of the albedo maps) being captured or computed for different types of skins. To solve this issue, we propose an approach suitable for *artistic editing* of subsurface scattering, based on physically meaningful parameters.

[3]Note that our pre-integrated formulation is fundamentally different from the one proposed by Penner and Borshukov [PB11]: while they pre-integrate the gradients in image-space due to subsurface scattering, we pre-integrate the *kernel*, which is later applied to simulate SSS.

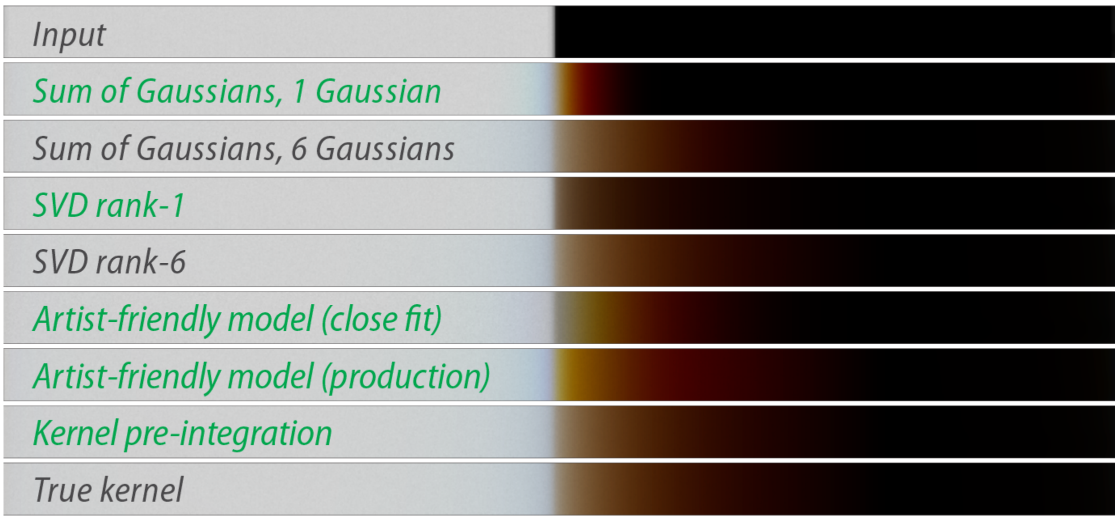


Figure 5.7: Comparison of the results for our two separable techniques applied to a step-like irradiance: both the sum-of-Gaussians [dLE07, JSG09] and a low-rank SVD-based decomposition need several convolutions to match the true kernel. Our proposed *pre-integrated* kernel, however, is exact for axis-aligned functions, such as this example. Our *artist-friendly* model provides a rich design space: This is illustrated by two approximations, where for the first (close fit), perceptual similarity to the ground truth was the modeling objective, while the second (production) was tweaked by an artist for production purposes. The style of presentation was inspired by previous work [DI11, HCJ13]. Additional 2D visualizations of the different kernels are provided in Section 5.7, where we also demonstrate that the radial asymmetry of the SVD-based kernels vanishes rapidly with increasing rank.

A general approximation adding separable kernels (see Equation 5.2) exhibits a vast amount of degrees of freedom if general functions a_i are used. Even a sum of N Gaussians (see Equation 5.3) requires the manipulation of $2N$ parameters per wavelength by the artist. Moreover, changing the parameters of one Gaussian in the presence of many others may lead to unexpected results. Jimenez et al.[JJG12] proposed ad-hoc transformations to a base profile in a separable approximation to overcome this limitation. However, this model lacked an intuitive mapping to the underlying physics of diffusion. Instead, inspired by previous work allowing modeling and editing of subsurface profiles with simple low-dimensional functions [KKCF13], our approach is based on the more intuitive concept of splitting subsurface scattering into near- and far-range scattering, encoded in two Gaussians. These form the basis of our separable kernel a_m as:

$$\begin{aligned} a_m(x) &= w\,G(x,\sigma_n) + (1-w)\,G(x,\sigma_f) \\ A_m(x,y) &= a_m(x)a_m(y), \end{aligned} \tag{5.6}$$

where σ_n and σ_f represent the standard deviation of the near and far scattering Gaussians respectively, and w is the weighting factor. Note that a_m represents a mixture of two 1D

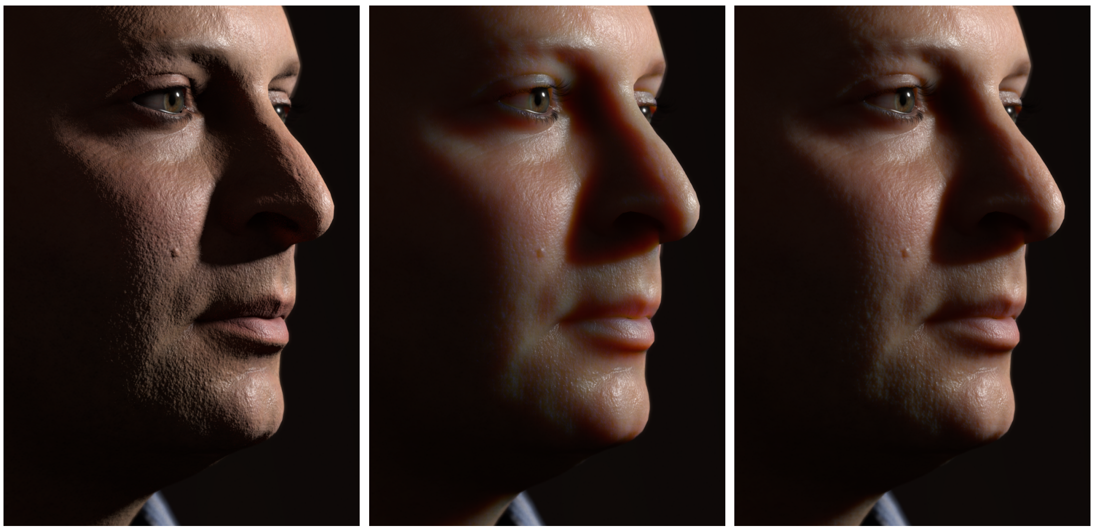


Figure 5.8: An example of the intuitive editing capabilities of our artist-friendly model. Left: Input irradiance map, without subsurface scattering. Middle: Adjusting the far scattering. Right: Final result after adjusting the near scattering and the balance between the two. Shifting more energy to near scattering allows preserving the bump details.

Gaussians $G(x, \sigma)$, which result in a separable (rank-1) kernel. This is different from the 2D Gaussian mixture approach of d'Eon et al. [dLE07], where two Gaussians would not yield a separable (rank-1) kernel, but a solution of rank two.

Our model takes the benefits of the separable approximation described in Section 5.3, which is able to match the ground truth under different light configurations (Section 5.4), while offering a rich design space that allows for intuitive editing of the *appearance* of translucent materials (Figures 5.8 and 5.39(d)), including easy integration in production pipelines, where this appearance control is usually required. Our approximation features another important property: since it is based on Gaussians, it is a *continuous* parametric representation of the profile; this representation allows computing the diffusion profile analytically at run-time, as will be shown in Section 5.10. Moreover, the following section introduces an additional *guided* deviation from the actual diffusion kernel, to generate viable rank-1 approximations that emphasize certain translucency effects.

5.6 Guided Optimization

In this section, we provide more details of the guided optimization and discuss the practical effects of our k parameter. For a guided deviation from the diffusion kernel, we additionally present a practical optimization framework to generate viable rank-1 approximations that emphasize the translucency effects of certain features. We aim to find a separable approximation $A_s(x, y) = a_s(x)a_s(y)$ to the diffusion profile $R_d(x, y)$, defined by the solution to the minimization problem:

$$a_s = \underset{a}{\operatorname{argmin}} \int_{\mathbb{R}^2} \Gamma(x,y)\big(R_d(x,y) - a(x)a(y)\big)^2 \, dx \, dy \quad \text{subject to } \|R_d\|_1 = \|a\|_1^2. \tag{5.7}$$

We optimize for minimal $\mathcal{L}_2$ distance while still retaining energy conservation via the 1-norm constraint (see Section 5.3). The *guide function* $\Gamma(x, y)$ provides the means to select the appropriate length scale of intended artistic effect, and has the form:

$$\Gamma(x, y; k) = \left(x^2 + y^2\right)^{k/2} (1 - e^{-bx^2})(1 - e^{-by^2}), \tag{5.8}$$

where k denotes the *guide parameter* and b gives the *suppression term* of the center cross region (we use $b = 50$). Varying k between 0 and 4 provides control over the perceived sharpness of the approximation A_s. $k \approx 2$ yields the approximation of the actual diffusion kernel, while $k = 0$ provides a visually sharper variant (see Figure 5.7) and $k = 4$ a smoother approximation that models the far-range scattering of the diffusion profile more faithfully. This optimization-based approach approximates diffusion kernels while still allowing to emphasize either near or far scattering with the help of a single parameter. This way, the user can adapt the kernel to a given scene and its specific properties, which is not possible by using the single, fixed solution of the pre-integration scheme.

5.7 Derivations and Details for Low-Rank Subsurface Light Transport

In this section, we provide more details about the low-rank approximation techniques, followed by a few important implementation details.

5.7.1 Low-rank approximations

In order to accurately approximate the diffuse reflectance profile $R_d(x, y)$ with a sum of separable kernels $A(x, y) = \sum_{i=1}^{N} a_i(x)a_i(y)$, an adequate choice for the individual 1D functions a_i is required. d'Eon and colleagues [dLE07] observed that zero-mean Gaussians G are suitable for this task, i.e.,

$$R_d(x, y) \approx A_g(x, y) = \sum_{i=1}^{N} w_i G(x, y; \tau_i), \tag{5.9}$$

where τ_i denote the standard deviations of the respective Gaussians. Due to the separability of the Gaussian kernel, the convolution with A_g can be realized as $2N$ 1D convolutions. During the computation of adequate w_i's and τ_i's, d'Eon et al. employed an $\mathcal{L}_1$ constraint such that $\|A_g\|_1 = \|R_d\|_1$, which guarantees energy conservation of the approximation A_g. Furthermore, the Gaussian kernel is spherically symmetric and thus A_g also exhibits this feature of the original profile.

We found, however, that emphasizing the closeness criterion and forfeiting radial symmetry yields approximations that produce the same visual quality with less convolutions, i.e., with lower complexity (see Figure 5.13). Inspired by previous methods for low-rank approximations of reflectance data [LRR04, PvBM+06, KM99], we employ matrix factorization of the *discrete* 2D diffusion profile $R_d \in \mathbb{R}^{m \times m}$. Following from the Eckhart-Young theorem [EY36], a truncation of the singular value decomposition (SVD) of R_d gives the best low-rank approximation with respect to the Frobenius norm. In more detail, given the SVD of the diffusion profile:

$$\begin{aligned} R_d &= U \Sigma V^T, \\ U &= \left(u^{(1)} | \, u^{(2)} | \, \ldots | \, u^{(m)} \right), \\ V &= \left(v^{(1)} | \, v^{(2)} | \, \ldots | \, v^{(m)} \right), \\ \Sigma &= \operatorname{diag}\left(\sigma_1, \sigma_2, \ldots, \sigma_m \right), \end{aligned} \tag{5.10}$$

the exact solution to the approximation problem

$$\begin{aligned} &\min_A \|R_d - A\|_F, \\ &\text{subject to } \operatorname{rank}(A) = N, \end{aligned} \tag{5.11}$$

is given by

$$\begin{aligned} A_s &= U \Sigma_N V^T, \\ \text{where } \Sigma_N &= \operatorname{diag}\left(\sigma_1, \ldots, \sigma_N, 0, \ldots, 0 \right). \end{aligned} \tag{5.12}$$

The Frobenius norm follows the classical definition, i.e., $\|A\|_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^m a_{ij}^2} = \sqrt{\sum_{i=1}^m \sigma_i^2}$ and corresponds to the $\mathcal{L}_2$ norm for continuous 2D functions. Note that Σ_N is a diagonal matrix and the approximation A_s can be written as a sum of separable kernels, i.e., $A_S = \sum_{i=1}^N u^{(i)} \, \sigma_i \, {v^{(i)}}^T$. The 1D functions a_i are therefore given by $a_i = \sqrt{\sigma_i} u^{(i)}$, since due to the symmetry of R_d, $u^{(i)} = v^{(i)}$.

While this approach does not preserve spherical symmetry and energy conservation, it provides optimal closeness in the sense of the $\mathcal{L}_2$ norm. For low-rank approximations starting from $N = 3$, the diffusion profile is more faithfully reconstructed, and the violation of both spherical symmetry and the $\mathcal{L}_1$ norm are not perceivable. Energy conservation can be enforced by scaling the approximation A_s according to $\tilde{A}_s = A_s \frac{\|R_d\|_1}{\|A_s\|_1}$. Although the new approximation $\tilde{A}_s$ is not optimal in the $\mathcal{L}_2$ sense, it still provides a perceptionally better match than the Gaussian approximation A_g with the same number of terms. Thus, our SVD-based approximation scheme yields better performance and the same or superior visual quality at the same number of 1D convolutions when compared to the Gaussian approximation (see Figure 5.13).

5.8 Derivation of the Pre-Integrated Approximation

Due to the non-separability of realistic diffusion profiles, it is not possible to fully reconstruct the effect of their convolution with 2D signals by a single separable kernel. It is, however, possible to completely reproduce a profile's behavior on a special class of signals. Below, we present an extended version of the derivation of Equation 5.4,

$$
\begin{aligned}
M_e(x,y) &= \iint E(x',y')\,R_d(x-x',y-y')\;dx'dy' \\
&= \iint (E_1(x') + E_2(y'))\,R_d(x-x',y-y')\;dx'dy' \\
&= \int E_1(x') \underbrace{\int R_d(x-x',y-y')\;dy'}_{a_p(x-x')}\,dx' \\
&\quad + \int E_2(y') \underbrace{\int R_d(x-x',y-y')\;dx'}_{a_p(y-y')}\,dy' \\
&= \int E_1(x')\,a_P(x-x') \underbrace{\frac{1}{\|a_p\|_1}\int a_p(y-y')\;dy'}_{=1}\,dx' \\
&\quad + \int E_2(y')\,a_P(y-y') \overbrace{\frac{1}{\|a_p\|_1}\int a_p(x-x')\;dx'}\,dy' \\
&= \iint E(x',y')\frac{1}{\|a_p\|_1}a_p(x-x')a_p(y-y')\;dx'dy'.
\end{aligned}
\tag{5.13}
$$

5.8.1 Motivation for the guided rank-1 approximation

Motivated by the high quality of the pre-integrated approximation, we tried to achieve a practical optimization framework that allows the user to emphasize either near or far scattering by manipulating only a single parameter, similar to the two-Gaussian approximation, but starting from an accurate fit to a diffusion profile. For a default parameter, the output should provide a 'neutral' approximation close to the analytically derived pre-integrated approximation.

Stating the associated optimization problem as

$$
\begin{gathered}
a_s = \underset{a}{\operatorname{argmin}} \int_{\mathbb{R}^2} \Gamma(x,y)\big(R_d(x,y) - a(x)a(y)\big)^2\,dx\,dy \\
\text{subject to } \|R_d\|_1 = \|a\|_1^2,
\end{gathered}
\tag{5.14}
$$

we derive a separable approximation $A_s(x,y) = a_s(x)a_s(y)$ to the diffuse reflectance profile $R_d(x,y)$. The *guide function* $\Gamma(x,y)$ provides the means to alter the result to provide the desired artistic effect. Since an arbitrary guide function would exhibit a huge amount of degrees of freedom, we aim to restrict its shape to a one-dimensional subspace that serves the intended effect of pronouncing either near or far scattering.

As a starting point, we tried to determine the guide function that would reproduce the pre-integrated approximation, i.e., we numerically calculated a guide function $\Gamma_p(x,y)$

such that

$$a_p = \underset{a}{\operatorname{argmin}} \int_{\mathbb{R}^2} \Gamma_p(x,y)\big(R_d(x,y) - a(x)a(y)\big)^2\,dx\,dy \qquad \text{subject to } \|R_d\|_1 = \|a\|_1^2, \tag{5.15}$$

where a_p is the pre-integrated approximation of the previous Section 5.8. As can be seen in Figure 5.9a, such a guide function exhibits a rather complicated structure. To reduce the associated complexity down to a single parameter, we empirically defined a parametrized guide function

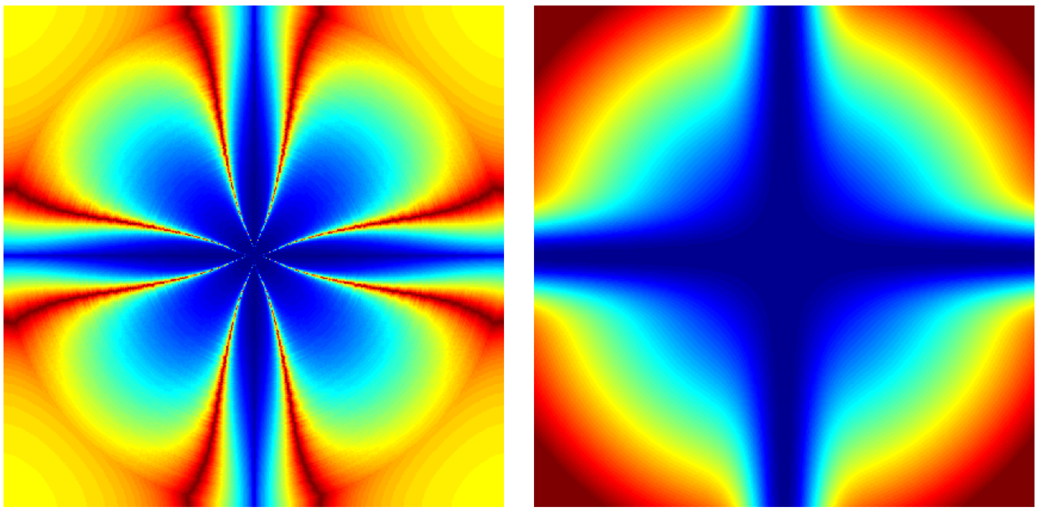
$$\Gamma(x,y;k) = \left(x^2+y^2\right)^{k/2}(1-e^{-bx^2})(1-e^{-by^2}). \tag{5.16}$$

The radially symmetric polynomial term with exponent k serves as external parameter to either emphasize near or far scattering. A depiction of this function can be found in Figure 5.9b for $k = 1.55$. Note that this polynomial term reproduces the behavior of $\Gamma_p(x,y)$ on the diagonals (while ignoring the minor normalization-dependent details in the corners). To reproduce the shape of Γ_p along the coordinate axes, we add 'suppression' functions in the form of $1-e^{-bx^2}$ and $1-e^{-by^2}$, which cause the guide function to vanish close to the coordinate axis. We chose $b = 50$ (assuming that $x, y \in [-1,1]$), and the whole framework is insensitive to moderate changes of this parameter. Varying the parameter k in the range from 0 to 4 allows the user to generate a separable approximation (as a result of Equation 5.14) that emphasizes near scattering (towards $k = 0$) or far scattering (towards $k = 4$). Note that no appreciable visual effects are observable beyond $k = 4$. We have used sequential quadratic programming to minimize Equation 5.14. To avoid taking on a high-dimensional optimization problem immediately, we solve Equation 5.14 for low-resolution versions of the final kernel. Each optimization is then initialized with an interpolation of the solution to the next-lower resolution.

5.8.2 Shader Implemention Details

As our technique works in screen space, the size of the kernel is a function of the projected surface area in the pixel, which depends on depth and surface orientation. This area is typically specified in world-space units instead of pixels, making the definition of the kernel size more intuitive for artists. When using a discretized kernel, the effect of the surface orientation can be taken into account by using ad-hoc correction factors [JSG09].

In contrast, using the simplified two-Gaussian artist-friendly model A_m allows us to work on real-world distances: we transform the depth of the pixel being evaluated and of each sample to world space, and then calculate the distance d between them. This distance is used to apply the profile on the fly, yielding more accurate results than using derivatives or ad-hoc correction factors. However, this approach has two problems: (1) we cannot accurately bake the kernel weights by evaluating the area covered by each sample (so we can only evaluate the kernel in the sample position), and (2) the number of slots per sample used by the GPU (as generated by DirectX 11 `fxc`) increases from 16 in a simple ad-hoc correction technique to 28, almost halving the performance, making it useless in



(a) $\Gamma_p(x, y)$ (b) $\Gamma(x, y; 1.55)$

Figure 5.9: Comparison of the guide function that reproduces the pre-integrated approximation (left) and our parametrized version that provides artistic control (right).

production scenarios. The latter problem has two reasons: a) converting from depth and pixel position to world-space requires a few additional ALU instructions; b) evaluating a 2-Gaussian RGB profile requires 6 `exp` instructions, which are extremely costly to execute even on modern GPUs.

We solve these problems by following an approach similar to the one proposed by Mikkelsen [Mik10], splitting the 1D profile application on 3D distances d to 2D distances d_{xy} with an accurate depth d_z correction factor. This transforms the evaluation of the kernel as:

$$\begin{aligned}
a_m\left(\sqrt{d_{xy}^2 + d_z^2}\right) &= w\,G\left(\sqrt{d_{xy}^2 \;+\; d_z^2}, \tau_{near}\right) + (1-w)\,G\left(\sqrt{d_{xy}^2 + d_z^2}, \tau_{far}\right) \\
&= w\,e^{\frac{-d_z^2}{2\tau_{near}}}\,G(d_{xy}, \tau_{near}) \;+\; (1-w)\,e^{\frac{-d_z^2}{2\tau_{far}}}\,G(d_{xy}, \tau_{far}) \\
&\approx e^{\frac{-d_z^2}{2\tau_{max}}}\Big(w\,G(d_{xy}, \tau_{near}) \;+\; (1-w)\,G(d_{xy}, \tau_{far})\Big) \\
&= e^{\frac{-d_z^2}{2\tau_{max}}}\,a'_m(d_{xy}),
\end{aligned}$$

where $\tau_{max} = max(\tau_{near}, \tau_{far})$. Note that we are making the approximation of taking the maximum variance of the Gaussians, which simplifies the profile application to an accurate depth correction, versus the ad-hoc corrections used in e.g. [JSG09]. This allows us to pre-compute accurate weights for $a'_m(d_{xy})$ using area integration, and reduce the number of instructions to 16 by: a) reducing the number of `exp` from 6 to a single one; b) avoiding the conversion to world space by directly working with depths; c) applying typical low-level optimizations [Per14].

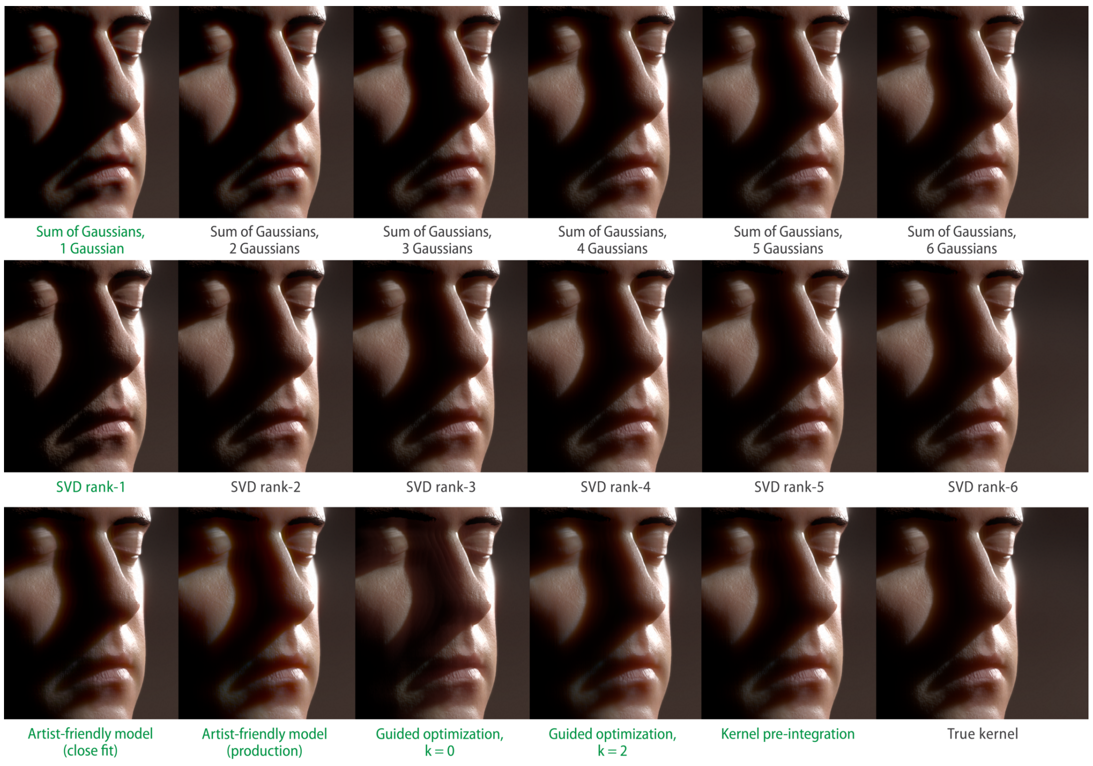

Sum of Gaussians, 1 Gaussian
Sum of Gaussians, 2 Gaussians
Sum of Gaussians, 3 Gaussians
Sum of Gaussians, 4 Gaussians
Sum of Gaussians, 5 Gaussians
Sum of Gaussians, 6 Gaussians
SVD rank-1
SVD rank-2
SVD rank-3
SVD rank-4
SVD rank-5
SVD rank-6
Artist-friendly model (close fit)
Artist-friendly model (production)
Guided optimization, k = 0
Guided optimization, k = 2
Kernel pre-integration
True kernel

Figure 5.10: Comparison of the different techniques proposed with d'Eon's method [dLE07] with six Gaussians, and the ground truth (actual 2D kernel). Note that our separable approximations lead to similar quality results with just two 1D convolutions, as opposed to the twelve needed by the sum of Gaussians approach. Note that all separable rank-1 kernels are highlighted with green.

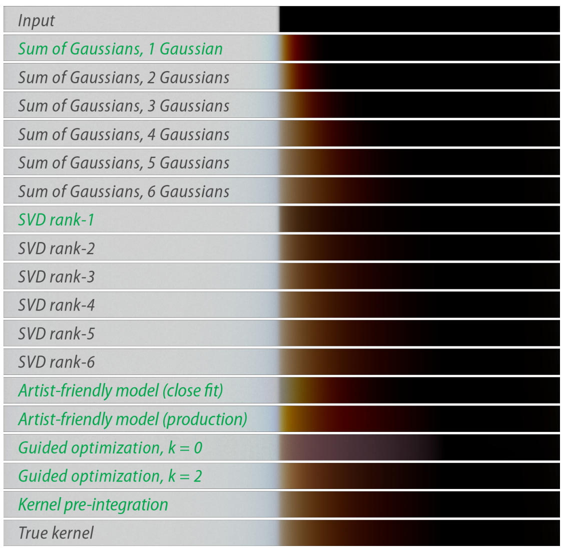


Figure 5.11: Our results reveal that SVD-based low-rank approximations scale better with the number of convolutions than the state of the art [dLE07], however, they still yield only a coarse approximation of the true kernel in the separable case (rank-1). Note that all separable rank-1 kernels are highlighted in green. The pre-integrated kernel is exact for axis-aligned functions, such as this example. The guided optimization with $k = 2$ provides a comparably good fit, while the $k = 0$ case captures the fine details near the boundary at the expense of the far-range scattering quality. A manual approximation using our artist-friendly model is illustrated by two approximations, where for the first (close fit), perceptual similarity to the ground truth was the modeling objective, while the second (production) was tweaked by an artist for production purposes.

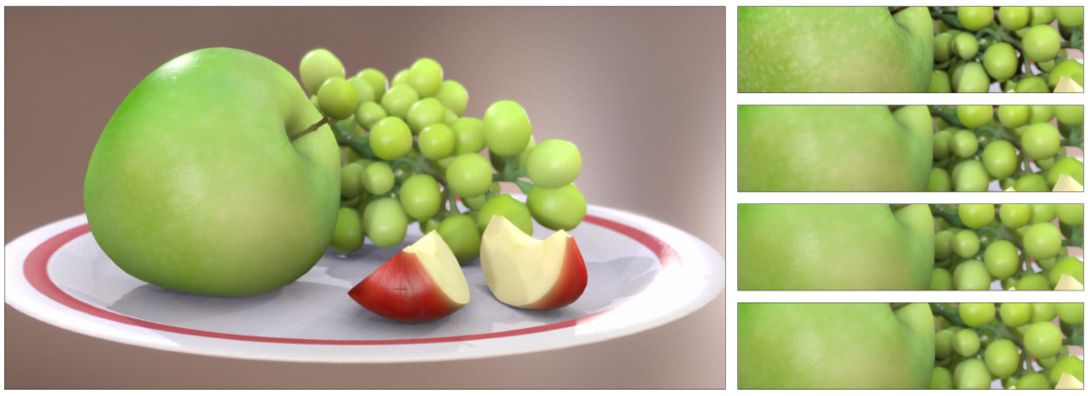


Figure 5.12: Real-time results for *apple*. The insets show (from top to bottom) input irradiance, d'Eon et al. [dLE07] with 1 Gaussian, our analytic kernel pre-integration technique and the ground truth. Both d'Eon's with one Gaussian and ours are run with the same number of convolutions, thereby yielding similar execution times.

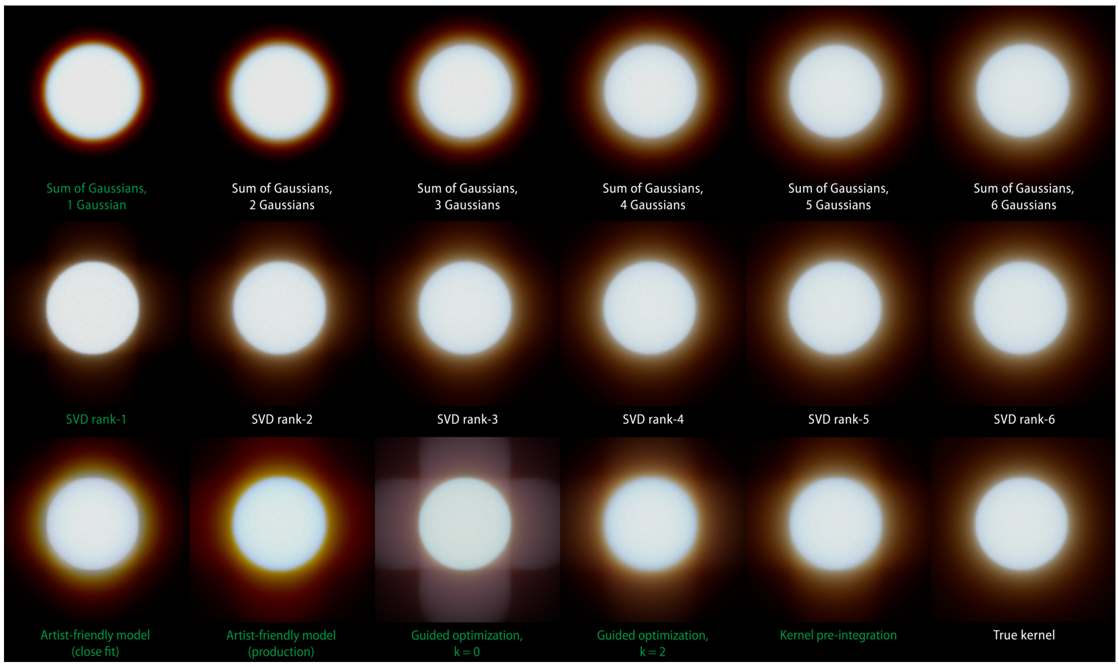
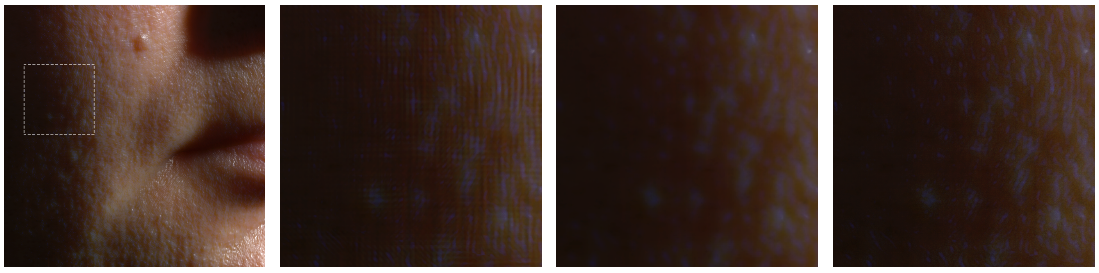

Sum of Gaussians, 1 Gaussian
Sum of Gaussians, 2 Gaussians
Sum of Gaussians, 3 Gaussians
Sum of Gaussians, 4 Gaussians
Sum of Gaussians, 5 Gaussians
Sum of Gaussians, 6 Gaussians
SVD rank-1
SVD rank-2
SVD rank-3
SVD rank-4
SVD rank-5
SVD rank-6
Artist-friendly model (close fit)
Artist-friendly model (production)
Guided optimization, k = 0
Guided optimization, k = 2
Kernel pre-integration
True kernel

Figure 5.13: Additional results reveal the inherent shape and quality of the separable and low-rank approximations on a white disk irradiance signal. Please note that, although the radial asymmetry of our separable kernels and the gradient-color differences of the manual approximations are noticeable in case of the artificial 'dot' illumination, these artifacts are less noticeable in case of our practical rendering examples.



(a) (b) 13 (Non-Jitt.) (c) 65 (Non-Jitt.) (d) 13 (Jittered)

Figure 5.14: In harsh lighting conditions, extreme close-ups may reveal artifacts even in the presence of importance sampling. (a) Initial image; (b) Importance sampling with 13 samples shows banding artifacts (please zoom in in the digital version for a better view); (d) Up to 65 samples are needed to eliminate visible banding; (c) Our jittering approach also eliminates banding while keeping the sample count low (this image is best viewed in the digital version).

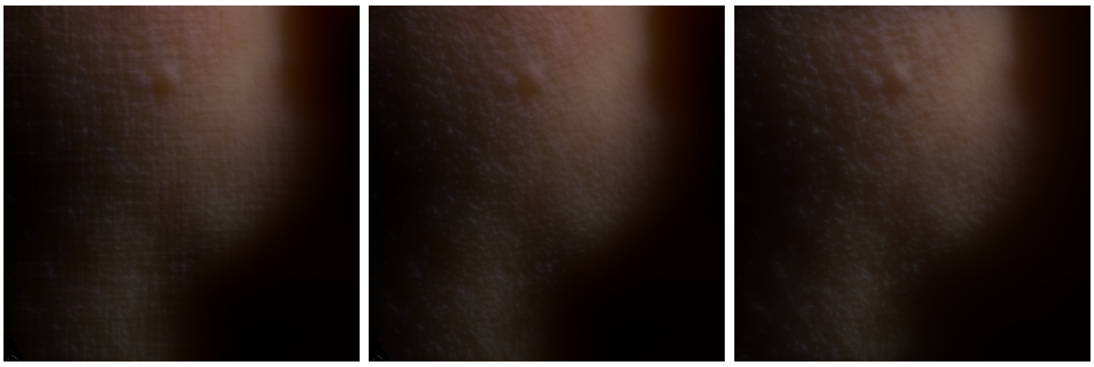

(a) Without jittered sampling (b) With jittered sampling (c) Ground truth

Figure 5.15: This figure illustrates that our jittered sampling scheme is able to remove banding artifacts stemming from the radial asymmetry of our separable kernels. The images represent our manual approximation (close fit) of human skin, which shows visible artifacts if no jittering is used (a), but is able to approximate the ground truth (b) in a visually plausible way if 33% of the samples are jittered (c).

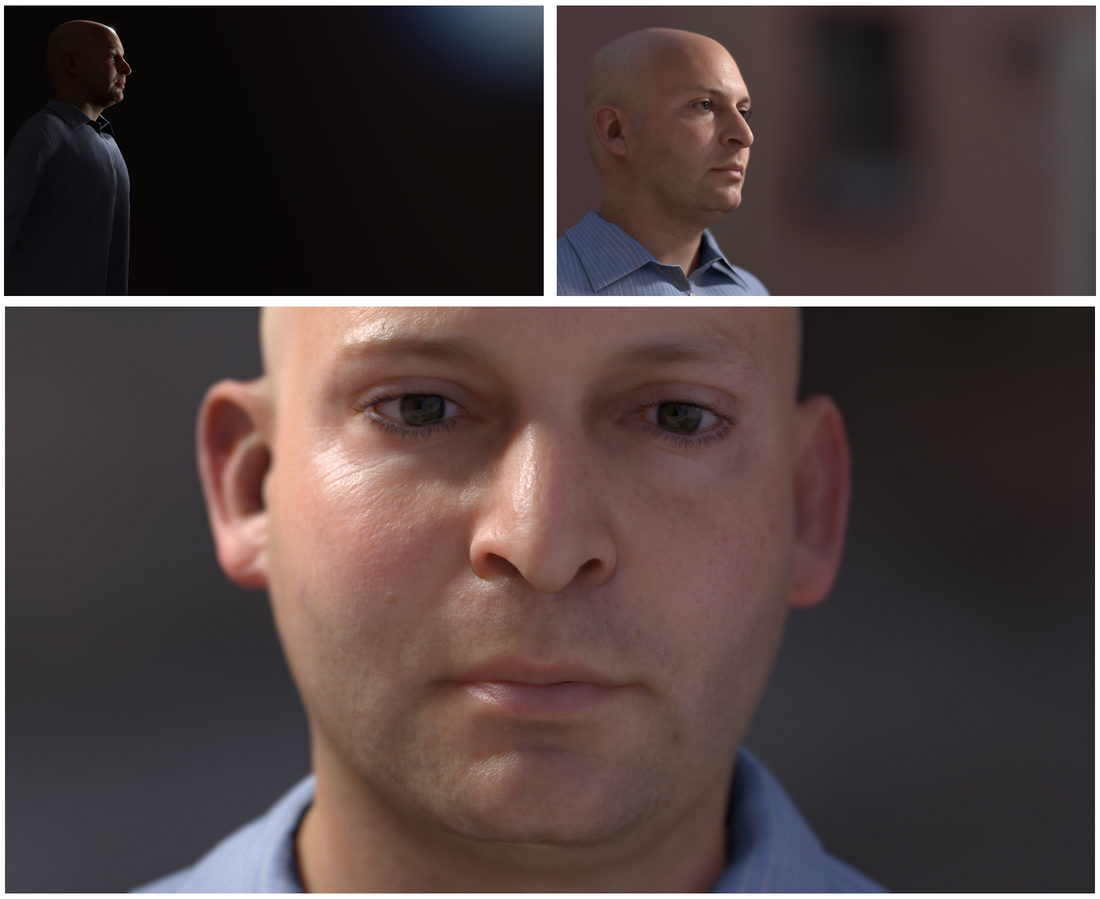


Figure 5.16: Typical cases found in games, for which the technique is designed.

5.9 Profile Simulation (MCML)

The diffuse reflectance profiles were simulated using MCML [WJZ95]. The material parameters used for our simulations were taken from an earlier work [JMLH01]. For each material, the RGB channels were simulated separately. Aside from the material properties, each channel used the same parameters specified as follows:

No. Photons: 10^7

Grid spacing: the minimum mean free path divided by 20 ($d_r = \frac{min(MFP_{rgb})}{20}$).

No. of grid elements: 32 times the maximum mean free path divided by d_r ($n_r = \lceil \frac{32 * max(MFP_{rgb})}{d_r} \rceil$).

Thickness: 10^7 cm (quasi-infinite).

The mean free path is computed as $\frac{1}{\sigma_t'} = \frac{1}{\sigma_a + \sigma_s'}$. As we model isotropic scattering ($g = 0$), the reduced scattering coefficient is trivial, i.e., $\sigma_s' = \sigma_s$. For the use with MCML, all parameters are also converted to cm, cm^{-1} or cm^{-2} respectively. MCML simulates cylindrically symmetric tissue models, and outputs the diffuse reflectance profile as a 1D function, $R_d(r)$. Detailed material parameters, plots of the simulated profiles as well as the derived radially symmetric 2D profiles are included in Section 5.9.1.

5.9.1 Simulation Parameters and Results

This section includes the properties of each measured material, simulated 1D profile plots and derived 2D profiles for each material. This gives a detailed description of the *input* to our method. The 1D profiles shown in the first two subplots are weighted by $2\pi r$ because the magnitude of the profile at a certain r value represents the reflectance on a complete circle of the radially symmetric 2D profile, whose area increases as r is increasing. It is worth noting that although all three channels were simulated up to the same maximum radius, the 1D profile plot may falsely suggest otherwise. In the case of the logarithmic scale plots, all profile values near machine precision are omitted by the plot function. In the MFP plots, all channel radii are scaled by the their respective MFP values, and therefore each channel has different support on the x axis. Additionally, all MFP plots show the unweighted profiles, and the 2D plots visualize only the most significant center region.

5.9.2 Apple

Apple	σ_a (mm^{-1}) [RGB]	σ_s (mm^{-1}) [RGB]	η	g	Thickness (cm)
Layer 1	[0.0030, 0.0034, 0.0460]	[2.2900, 2.3900, 1.9700]	1.3000	0.0000	10^7

Figure 5.17: Apple - Scattering properties

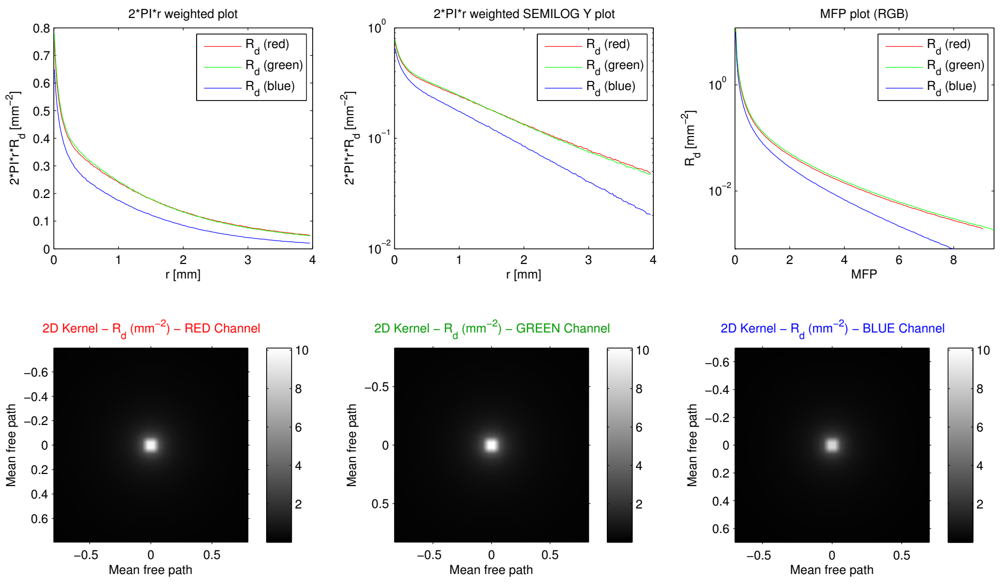


Figure 5.18: Apple - Simulation

5.9.3 Chicken1

Chicken1	σ_a (mm^{-1}) [RGB]	σ_s (mm^{-1}) [RGB]	η	g	Thickness (cm)
Layer 1	[0.0150, 0.0770, 0.1900]	[0.1500, 0.2100, 0.3800]	1.3000	0.0000	10^7

Figure 5.19: Chicken1 - Scattering properties

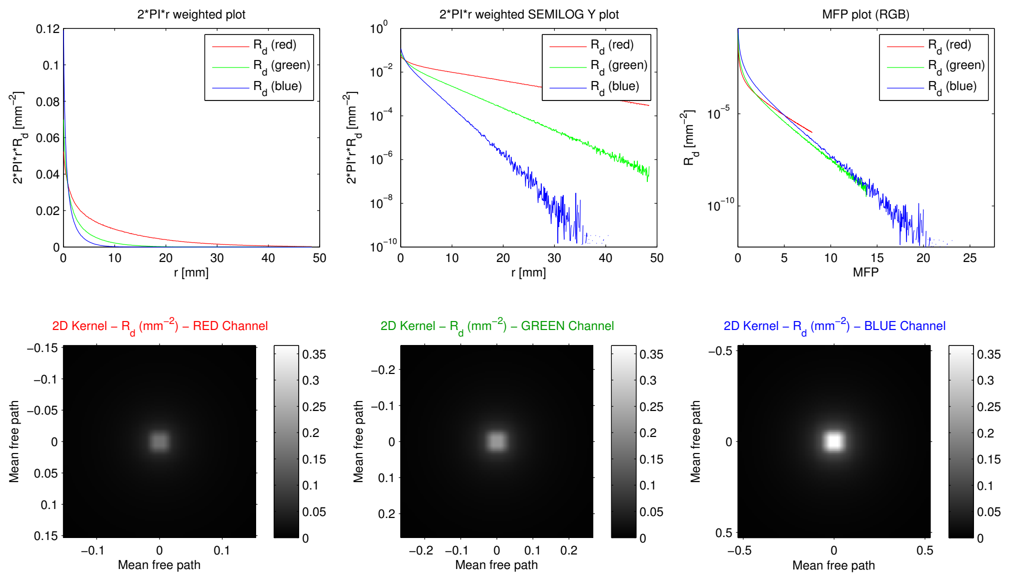


Figure 5.20: Chicken1 - Simulation

5.9.4 Chicken2

Chicken2	σ_a (mm^{-1}) [RGB]	σ_s (mm^{-1}) [RGB]	η	g	Thickness (cm)
Layer 1	[0.0180, 0.0880, 0.2000]	[0.1900, 0.2500, 0.3200]	1.3000	0.0000	10^7

Figure 5.21: Chicken2 - Scattering properties

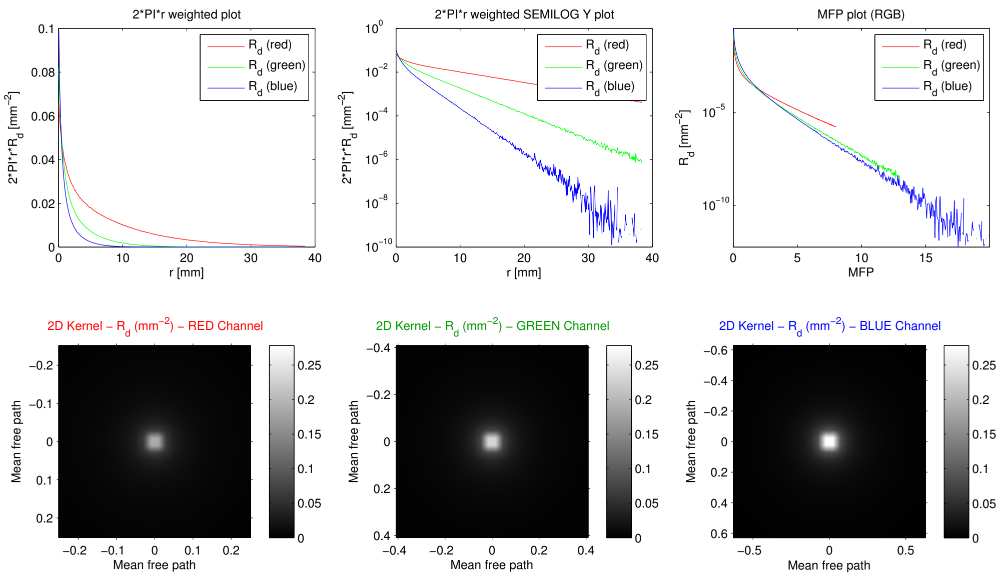


Figure 5.22: Chicken2 - Simulation

5.9.5 Cream

Cream	σ_a (mm^{-1}) [RGB]	σ_s (mm^{-1}) [RGB]	η	g	Thickness (cm)
Layer 1	[0.0002, 0.0028, 0.0163]	[7.3800, 5.4700, 3.1500]	1.3000	0.0000	10^7

Figure 5.23: Cream - Scattering properties

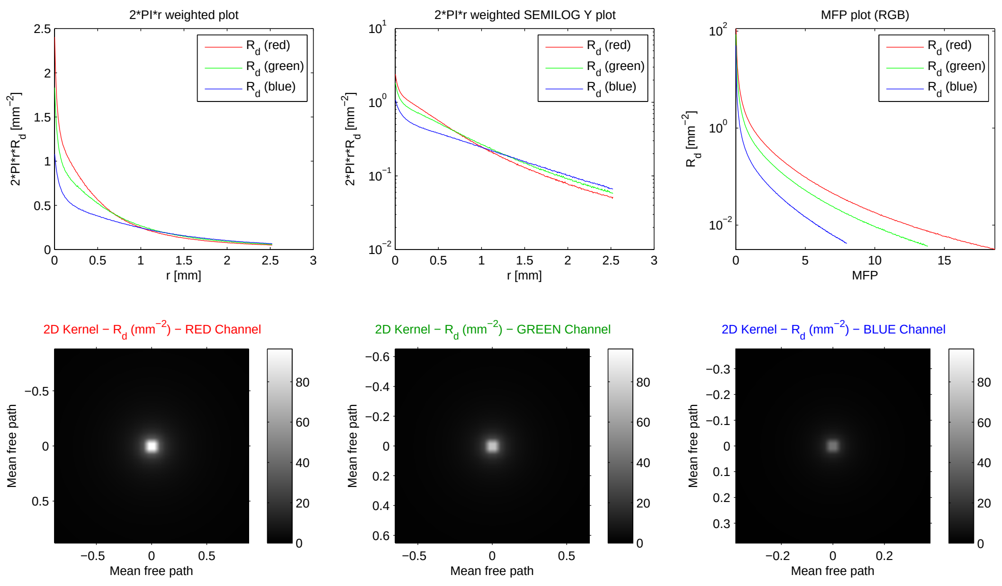


Figure 5.24: Cream - Simulation

5.9.6 Ketchup

Ketchup	σ_a (mm^{-1}) [RGB]	σ_s (mm^{-1}) [RGB]	η	g	Thickness (cm)
Layer 1	[0.0610, 0.9700, 1.4500]	[0.1800, 0.0700, 0.0300]	1.3000	0.0000	10^7

Figure 5.25: Ketchup - Scattering properties

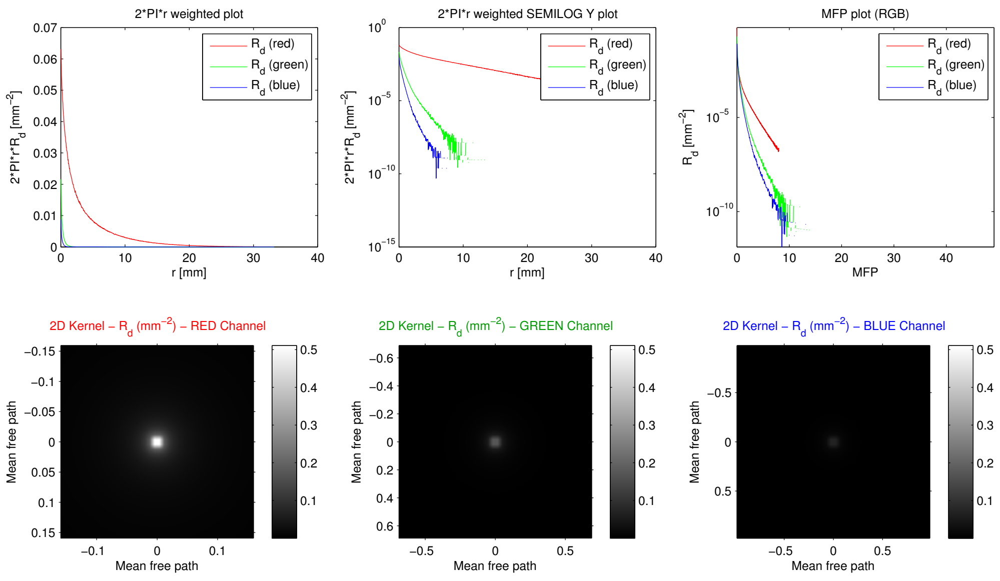


Figure 5.26: Ketchup - Simulation

5.9.7 Marble

Marble	σ_a (mm^{-1}) [RGB]	σ_s (mm^{-1}) [RGB]	η	g	Thickness (cm)
Layer 1	[0.0021, 0.0041, 0.0071]	[2.1900, 2.6200, 3.0000]	1.5000	0.0000	10^7

Figure 5.27: Marble - Scattering properties

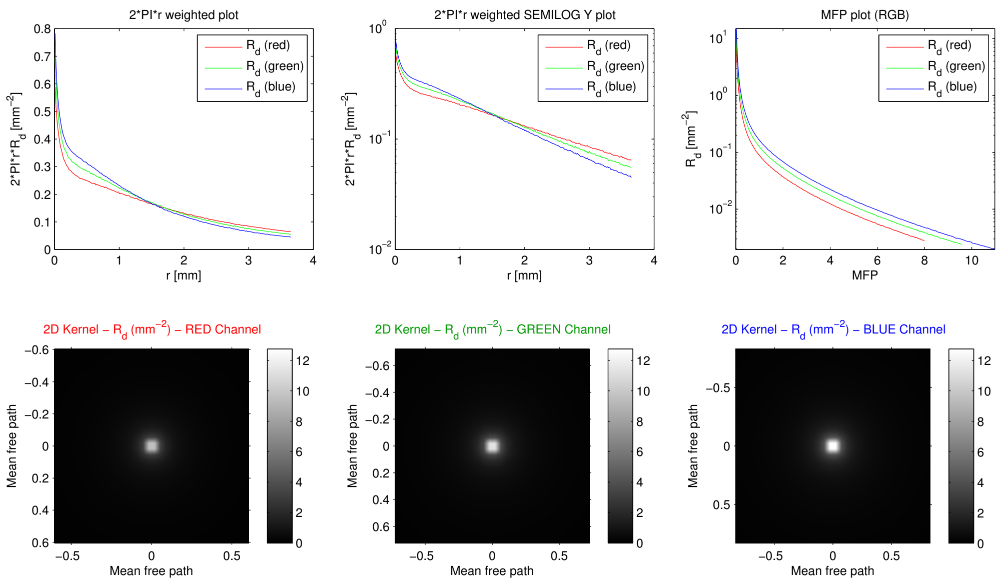


Figure 5.28: Marble - Simulation

5.9.8 Potato

Potato	σ_a (mm^{-1}) [RGB]	σ_s (mm^{-1}) [RGB]	η	g	Thickness (cm)
Layer 1	[0.0024, 0.0090, 0.1200]	[0.6800, 0.7000, 0.5500]	1.3000	0.0000	10^7

Figure 5.29: Potato - Scattering properties

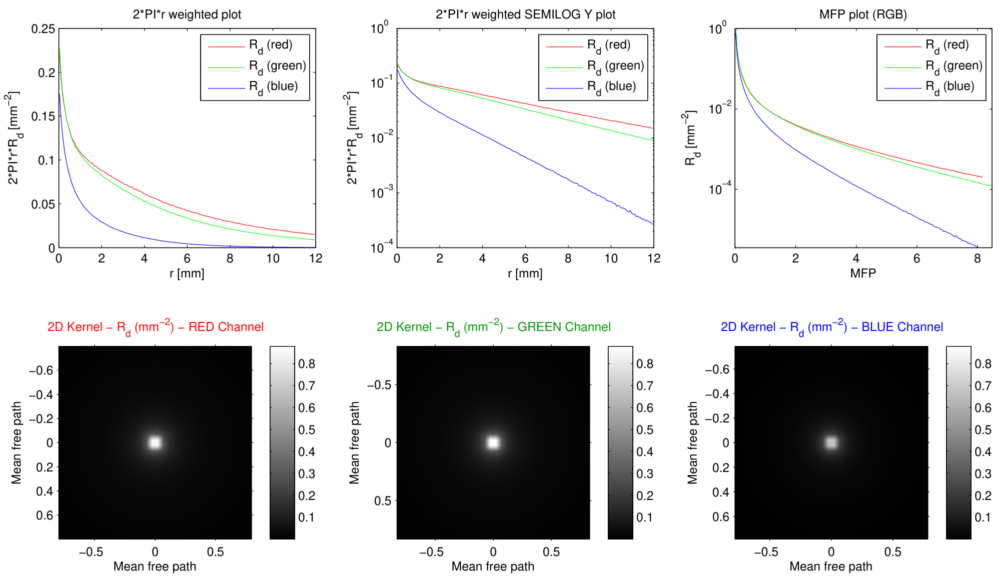


Figure 5.30: Potato - Simulation

5.9.9 Skimmilk

Skimmilk	σ_a (mm^{-1}) [RGB]	σ_s (mm^{-1}) [RGB]	η	g	Thickness (cm)
Layer 1	[0.0014, 0.0025, 0.0142]	[0.7000, 1.2200, 1.9000]	1.3000	0.0000	10^7

Figure 5.31: Skimmilk - Scattering properties

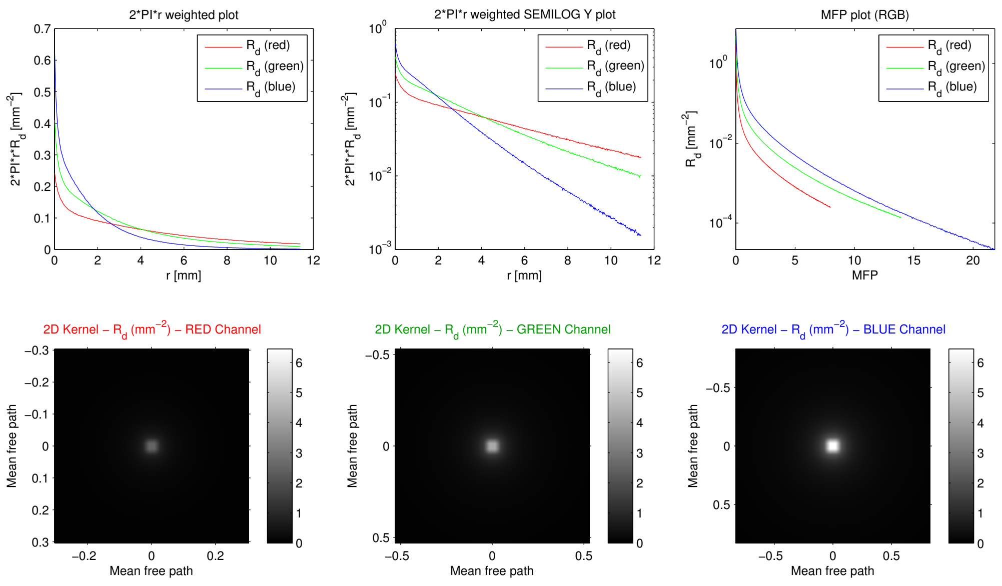


Figure 5.32: Skimmilk - Simulation

5.9.10 Skin1

Skin1	σ_a (mm^{-1}) [RGB]	σ_s (mm^{-1}) [RGB]	η	g	Thickness (cm)
Layer 1	[0.0320, 0.1700, 0.4800]	[0.7400, 0.8800, 1.0100]	1.3000	0.0000	10^7

Figure 5.33: Skin1 - Scattering properties

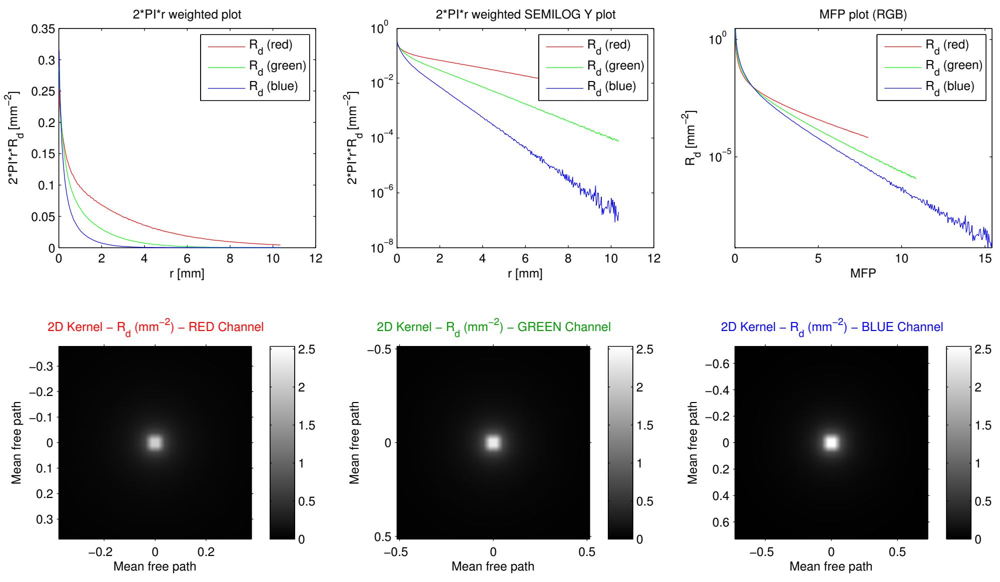


Figure 5.34: Skin1 - Simulation

5.9.11 Skin2

Skin2	σ_a (mm^{-1}) [RGB]	σ_s (mm^{-1}) [RGB]	η	g	Thickness (cm)
Layer 1	[0.0130, 0.0700, 0.1450]	[1.0900, 1.5900, 1.7900]	1.3000	0.0000	10^7

Figure 5.35: Skin2 - Scattering properties

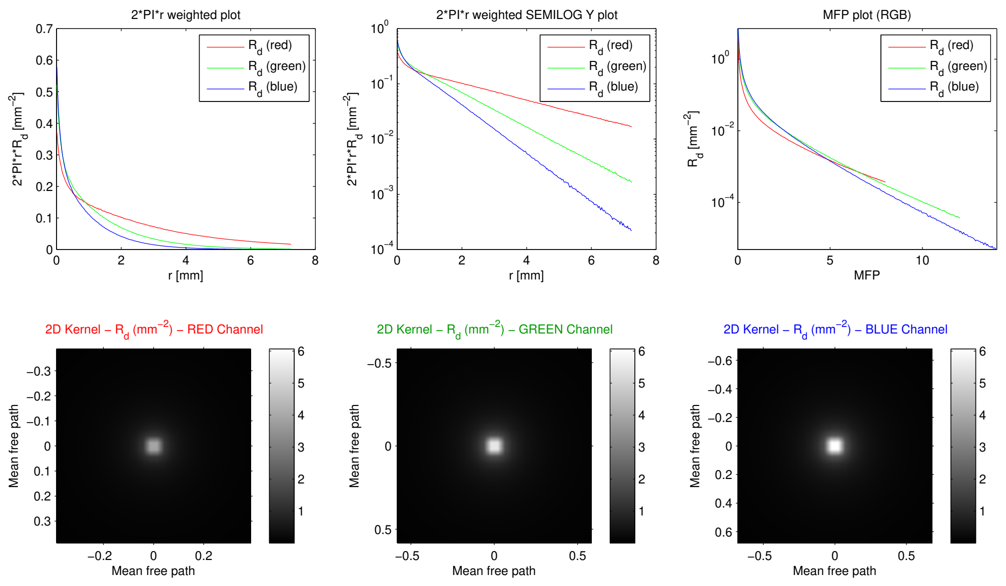


Figure 5.36: Skin2 - Simulation

5.9.12 Wholemilk

Wholemilk	σ_a (mm^{-1}) [RGB]	σ_s (mm^{-1}) [RGB]	η	g	Thickness (cm)
Layer 1	[0.0011, 0.0024, 0.0140]	[2.5500, 3.2100, 3.7700]	1.3000	0.0000	10^7

Figure 5.37: Wholemilk - Scattering properties

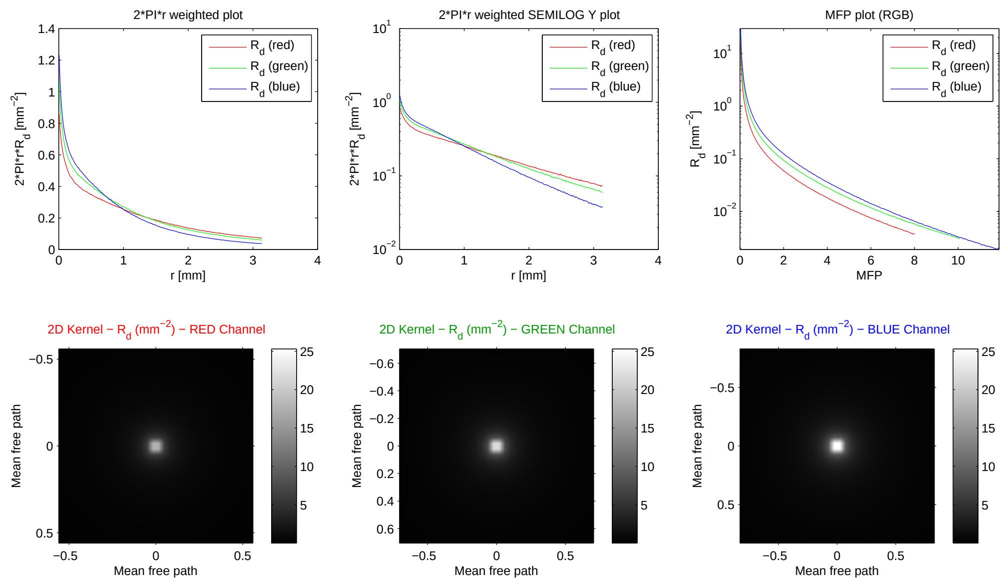


Figure 5.38: Wholemilk - Simulation

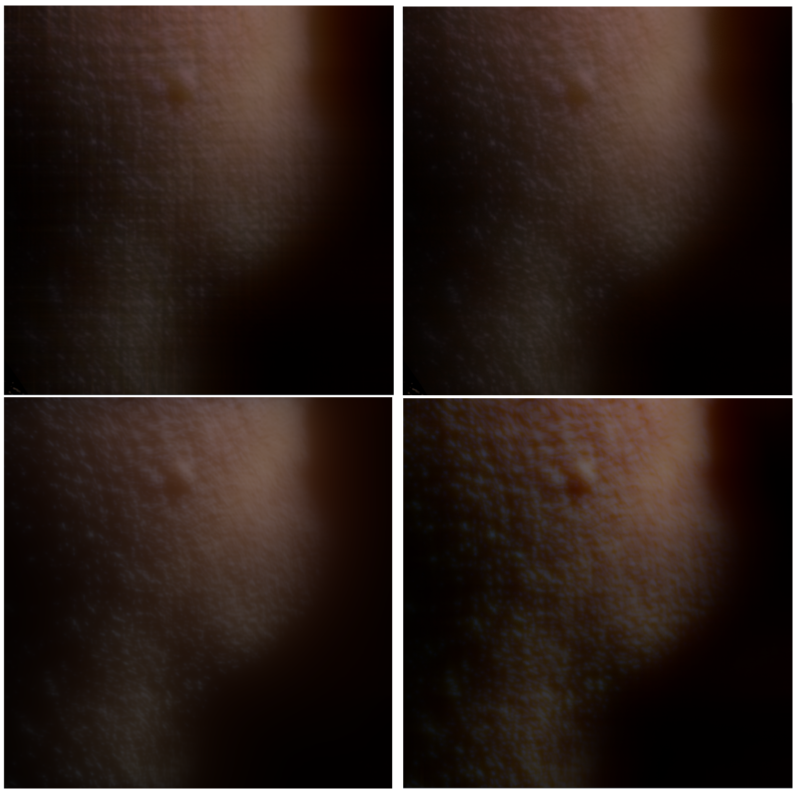


Figure 5.39: Although jittering for a radius of 10% of the kernel size is enough for usual portrait distances and resolutions of 1080p, extreme close-ups may reveal artifacts due to the non-separability of the kernel in these conditions (a). Performing jittering on a radius of 30% completely removes these artifacts (b), visually matching the result of the ground truth kernel (c). Also note that a kernel designed to match this particular asset using our artist-friendly model (d) shows a more natural look (this image is best viewed in the digital version).

5.10 Rendering

Our approximation of the diffusion profile, which is represented as just one separable kernel, can be applied both in texture- and in screen space. For efficiency, we use the screen-space approach of Jimenez et al. [JSG09], including translucency [JWSG10], separating into different buffers the albedo, diffuse and specular components, simulating subsurface transport only in the diffuse layer, and compositing for final rendering. In the following, we highlight additional improvements to the rendering pipeline, resulting in an optimized code of just 16 instructions per sample.

Jittering. Our separable approximations may lead to some artifacts under high-frequency illumination, as shown in Figure 5.40. This is because the spatial footprint of

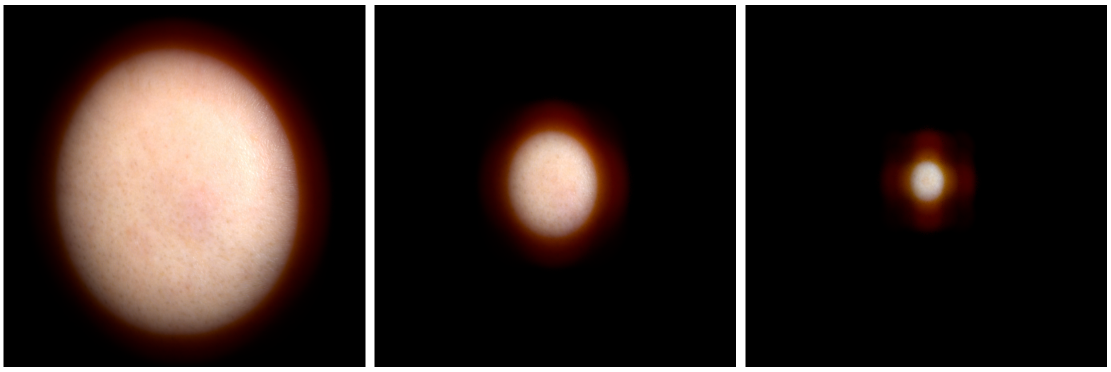


Figure 5.40: In harsh lighting conditions, progressively smaller features may reveal the fact that the applied separable profiles are not radially symmetric.

the signal becomes smaller than the bandwidth of the kernel, producing an asymmetric *star-like* pattern. This situation is common also in high-quality, close-up shots, showing for example the pores of skin in detail.

To alleviate this problem, we apply a randomized per-pixel rotation of the filtering axes, similar to Huang et al. [HBR$^+$11]. Using a randomized rotation, as opposed to other alternatives such as kernel jittering, has two key advantages: *(i)* it breaks the visible cross pattern; and *(ii)* due to the radial symmetry of the diffusion kernel, it is not necessary to reintegrate the kernel, since distances are preserved. To avoid GPU cache thrashing, we only apply this to samples close to the pixel being evaluated (closer than 10% of the kernel size in our implementation, although it is dependent on the zoom and the used kernel; for extreme close-ups and kernels modeled with very different Gaussian lobes, a higher range would be needed, as shown in Figure 5.39). This solves the artifacts in small-scale features such as skin pores, although for higher-scale features, such as the light dot in Figure 5.40, there may still be visible artifacts (note however that this is a pathological case, not common in real-world applications). Figure 5.39 demonstrates the effects of randomizing the sample positions. Note that, in addition to masking the artifacts, it also reduces banding problems due to under-sampling.

Kernel footprint and evaluation. For the screen-space method, the size of the convolution kernel is a function of the projected surface area of the pixel [JSG09]. For the continuous approximation (Section 5.5), we follow similar derivations as Mikkelsen [Mik10]: these allow *(i)* applying the kernel accurately in world-space, as opposed to using ad-hoc correction factors [JSG09], and *(ii)* computing the exact areas for each sample offline, while *(iii)* still being fast enough for demanding real-time applications.

Importance sampling. To compute the convolution with the 1D functions of the approximation (e.g., a_p in Equation 5.5) during rendering, they need to be discretized. In general these functions exhibit a very uneven energy distribution, so a uniform

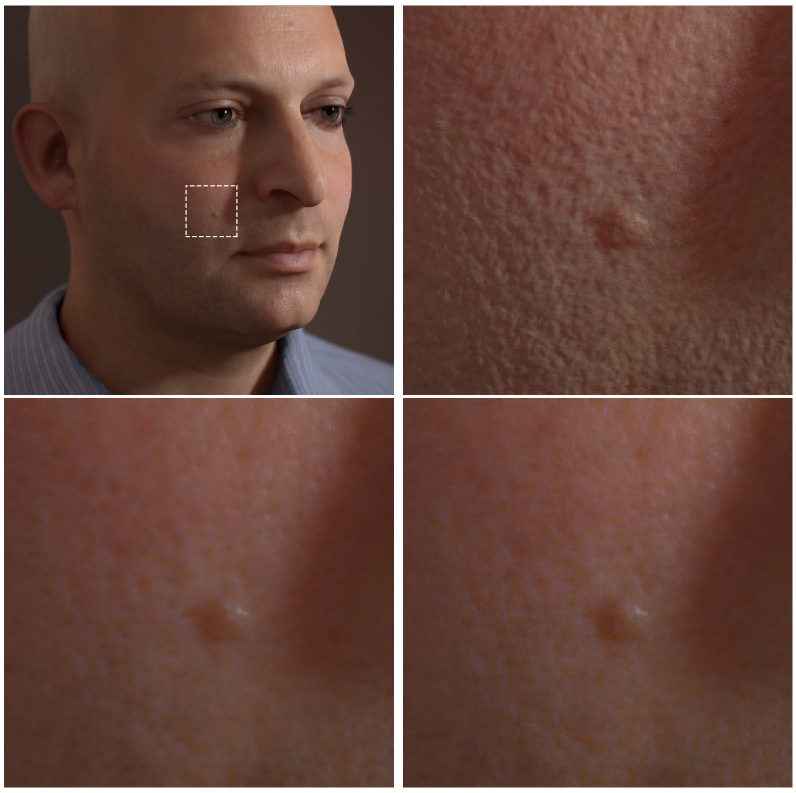


Figure 5.41: The quality of the rendered images, with and without importance sampling of the approximated kernel. (a) Initial image; (b) Uniform sampling with only seven samples leads to obvious aliasing artifacts; (c) Ground-truth created by uniform sampling with 513 samples; (d) Our importance sampling with as low as seven samples.

discretization will either require a high resolution for acceptable quality, which entails a significant performance impact, or result in aliasing when a lower resolution is used. To solve this issue, we use importance sampling on the 1D function by allocating a greater amount of sample points near the center, where most of the energy of the signal is found.

In order to minimize the execution time and the number of memory accesses, we sample all channels at the same positions determined by the dominant channel. In Figure 5.41, we demonstrate the difference in image quality with importance sampling compared to the ground truth and uniform sampling. In this particular portrait shot under natural lighting, using 7 samples with importance sampling allows a faithful representation of skin subsurface scattering. This importance sampling can also be used to improve the quality of previous methods, such as the sum-of-Gaussians approach [dLE07, JSG09].

5.11 Results

We validate our proposed techniques with a range of different translucent materials. Figures 5.1 (left), 5.6, and 5.42 depict *human skin*. Figure 5.7 compares between the

Kernel	σ_n [RGB]	σ_f [RGB]	w
Close fit	[0.034, 0.029, 0.016]	[1, 0.48, 0.2]	0.37
Production	[0.25, 0.08, 0.02]	[1, 0.3, 0.05]	0.6

Table 5.2: Parameters for our artist-friendly separable model with two different fits illustrated in our results.

different methods also using the skin diffusion profile. Figure 5.43 features results for *soap* and *marble*, while Figure 5.1 (right) shows *ketchup*.

We note that the execution time of every technique described in this manuscript is dominated by the convolution computation, so their timings are roughly proportional to the number of convolutions they require (i.e., six Gaussians is approximately six times slower than our separable approximation). Figure 5.6 compares between our separable approximation, the current state-of-the-art [dLE07, JSG09], and the ground-truth 2D convolution: our pre-integrated kernel gives visually convincing results and can be computed analytically from an arbitrary profile. This enables subsurface scattering to be viable for severely time-constrained real-time applications. Note that in the case of a one-dimensional irradiance signal, such as the one in Figure 5.7, the pre-integration technique yields the *analytic* ground truth.

While the artist-controlled approximation allows designing the diffusion arbitrarily, it can also be used to match a specific profile manually; although not as exact as the pre-integrated technique, it still captures key features in a visually convincing way, such as color bleeding into the shadowed region, and it can be seamlessly used in production scenarios. The close-fit kernel for human skin (Figures 5.7, 5.6 and 5.39(b,c)) shows how the two-Gaussian artistic model can be used to approximate the ground truth. We again emphasize that this two-Gaussian approach is different from the sum of Gaussians proposed by d'Eon et al. To demonstrate the editing capabilities of our artist-friendly model, we show results using different types of skin, including one actually developed by an artist for improving the results in our particular assets, and another trying to mimic the exact skin diffusion profile (Figure 5.7), and that we can adapt the appearance of translucency to the goal asset, as seen in Figure 5.39(d). The parameters for both kernels are included in Table 5.2.

Our general convolution scheme can be implemented in both texture- and screen space. In the latter, the size of the kernel is a function of the projected surface area in the pixel [JSG09]. As this leads to a favorable fixed-cost property that is sought-after in real-time applications, we believe these techniques offer a desirable choice for such scenarios. Our method does not require any additional considerations for dynamic objects and moving light sources, and scales well with the area (0.498, 0.348 and 0.304 ms at 1080p on an AMD Radeon HD 7970 for a close-up, middle and long shot respectively). The quality of the simulation can be adjusted by setting the number of samples in the kernel, which gives a trade-off between quality and cost.

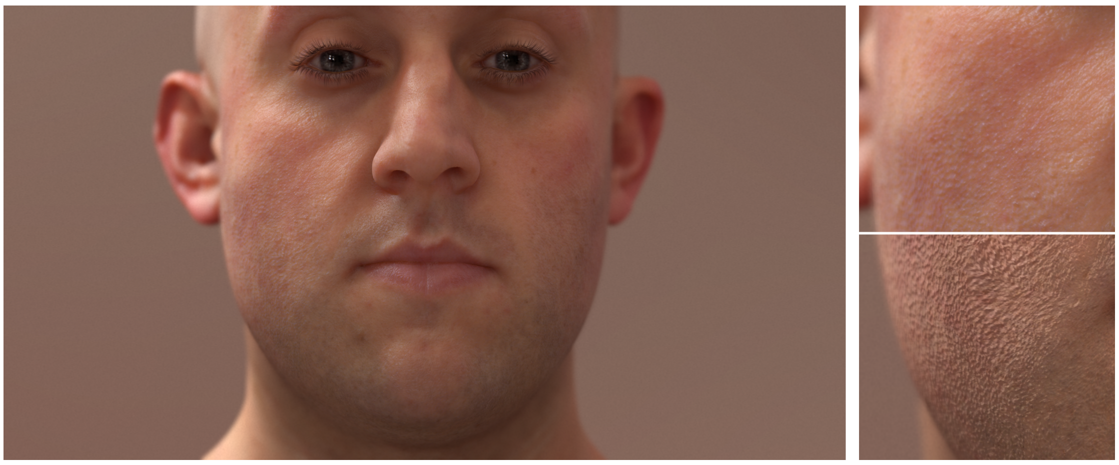


Figure 5.42: A close-up comparison before (bottom right) and after (top right) applying our separable subsurface-scattering model. Note that without taking into account subsurface light transport, the realism of the image is lost.

Limitations and future work. As discussed previously, trading off radial symmetry for separability might create some artifacts revealing the inherent shape of the kernel in extreme close-up views with high-frequency illumination. While the jittering proposed in Section 5.10 generally solves this issue for very small-scale details (e.g., skin pores), some artifacts may still remain in some situations (Figure 5.40). However, we have not experienced this phenomenon in any of the practical cases.

Section 5.3 shows how the error of our reconstruction is determined by the squared sum of the higher-order singular values. Our method relies on taking advantage of the fact that even though the diffusion profiles are of moderately high rank, the information contained therein is highly structured. Using our separable model, we have been able to match the target profiles well in terms of RMS error.

Since the pre-integrated approximation yields a faithful reconstruction of the diffusion profile with only one separable convolution, a low-rank approximation can be found by approximating the difference between the pre-integrated kernel and the diffusion profile. This could lead to better low-rank approximations. In our use cases, a low-rank SVD-based reconstruction already provided sufficient quality for ranks equal or larger than three (we have shown results in Section 5.7.). This suggests that an SVD-based reconstruction of diffusion kernels might be very useful for simulating efficient high-quality subsurface scattering in off-line or interactive environments.

5.12 Conclusions

We have presented two techniques to generate separable approximations of diffuse reflectance profiles to simulate subsurface scattering for a variety of materials using

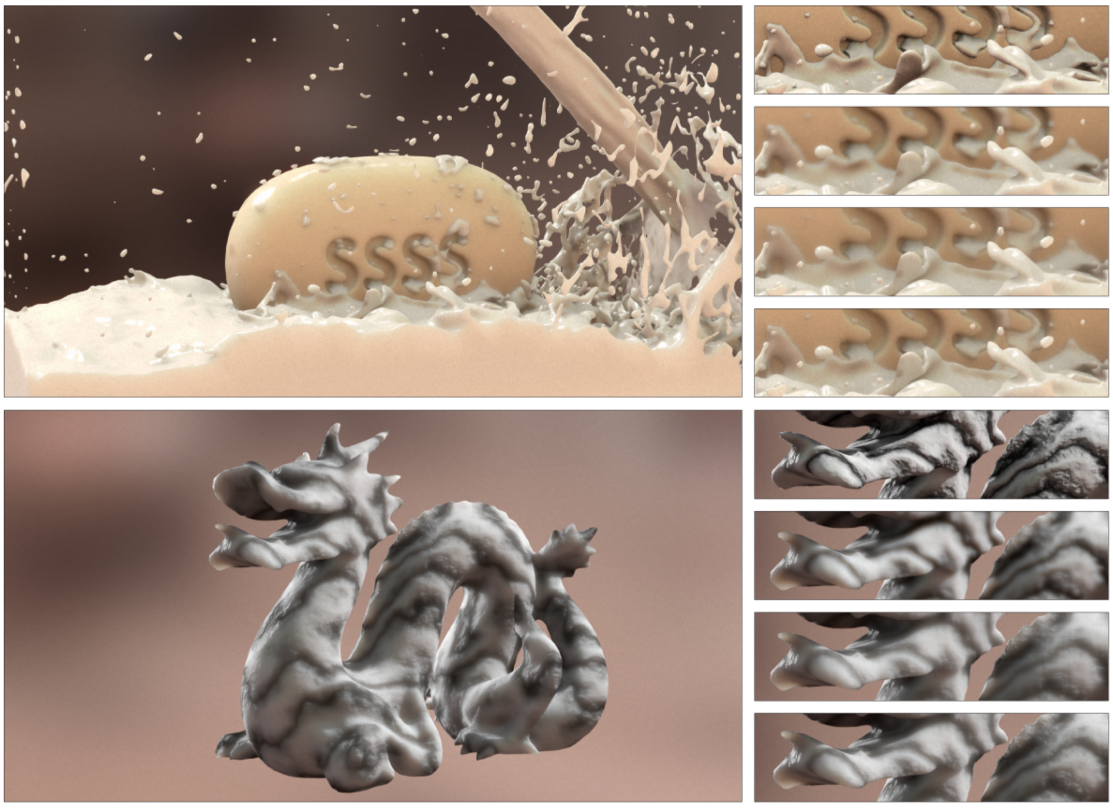


Figure 5.43: Real-time results for *soap* and *marble*. The insets show (from top to bottom) input irradiance, the sum-of-Gaussians approach [dLE07] with 1 Gaussian, our analytic kernel pre-integration technique and the ground truth. Both the sum of Gaussians and ours are run with the same number of convolutions, thereby yielding similar execution times. Our method is able to faithfully capture the effect of subsurface scattering in shadowed regions (*soap*), and retains intricate texture features (*dragon*).

just two 1D convolutions. Our separable models yield state-of-the-art results in less than 0.5 millisecond per frame, which makes high-quality subsurface scattering affordable even in the most challenging real-time contexts such as games, where every desired effect may have a budget of tenths of a millisecond.

Using axis-aligned pre-integration, we have presented a high-quality separable approximation that is provably optimal for additively separable irradiance signals. We also proposed an artist-friendly model that allows intuitive artistic control on the appearance of subsurface scattering based on only three parameters, and that allows seamless integration into our separable framework. We have additionally shown that low-rank approximations based on matrix factorization yield higher performance than the sum of Gaussians, which suggests an interesting avenue of future work for efficient subsurface scattering simulation.

Our algorithm works as a post-processing step, which makes it very efficient and simple to integrate in existing rendering pipelines, reducing the complex subsurface light transport [JMLH01] to its barebones (a seven samples blur filter with 16 assembly instructions per sample). Moreover, we have shown how combining importance sampling and jittering strategies allows using only seven samples per pixel in many cases of practical interest.

CHAPTER 6

Concluding Remarks

In this chapter, we provide a short summary of the main contributions of our projects.

6.1 Summary

In this thesis, we set out to improve the realism of modern rendering systems by improving the process of synthesizing photorealistic material models and proposed a technique to improve the state of the art in real-time subsurface light transport. Our first material synthesis method, **Gaussian Material Synthesis**, enables novice users to perform material synthesis without engaging in direct interaction with a "principled" shader that typically requires significant expertise in material modeling. This method starts by presenting the user with a gallery, and proceeds to generate an arbitrary amount of high-quality material models that are in line with their preferences. We also proposed a neural renderer that is able to mimic a light transport simulation program and visualize these material samples faithfully within a few milliseconds, thereby opening up the possibility of exploring these material models in real time. To further accentuate the advantages of this property of our method, we also proposed a learning-based technique that helps visualizing variants of a target material – this leads to a novel workflow where the artist can rapidly fine-tune a chosen material in real time without any expertise in photorealistic material synthesis. When more than a handful of materials are sought, even expert users are expected to see a speedup in the creation of these materials.

Our second method, **Photorealistic Material Editing Through Direct Image Manipulation**, offers a way for artists to reuse their general image-processing expertise to create sophisticated photorealistic material models. This process starts with the user applying well-known transformations (e.g., colorization, image inpainting) to a target image in a 2D raster editor, resulting in a non-physical image that often cannot be achieved by means of photorealistic rendering. However, our key observation is that in many cases, solutions can be found that are remarkably close to this target image. In order to find these solutions, we proposed a neural network that offers an initial guess that is further refined by our optimizer. This yields a simple and robust method that executes within 30 seconds, supports image sequences and works even in the presence of poorly edited images.

Finally, in **Separable Subsurface Scattering**, we have presented a practical method to simulate subsurface light transport for a variety of translucent materials by using only two convolutions. This technique yields state-of-the-art results in less than 0.5 milliseconds per frame, which makes it affordable even in the most challenging real-time contexts, such as modern games, where every desired effect has at most a budget of a few milliseconds. Our technique is based on representing the simulated diffuse reflectance profile by a separable kernel, can be performed through as few as two convolutions, where the established sum-of-Gaussians approximation would require many more passes to achieve the same visual quality, which leads to a significant reduction of both execution time and memory consumption. Since the algorithm is introduced as a post-processing step, it is remarkably simple to integrate into existing rendering pipelines, is suitable for challenging real-time applications and is practically free on modern graphics hardware. Animation and dynamic objects are also supported without any additional overhead or further changes. Moreover, we have offered a simplified approach, suitable for artistic editing of scattering profiles, and have shown how combining importance sampling and jittering strategies allows, in some cases, using only seven samples per pixel.

Creating beautiful photorealistic images and animations requires a high-quality rendering system that supports many of the most important light-transport phenomena appearing around us. However, on the other hand, we also have to be vigilant of the fact that these systems are used by people that are adept at carrying out their artistic vision, but do not have a rigorous understanding of the many physical quantities we use in these simulations. Hence, in this thesis, we endeavored to not only push the realism and efficiency of subsurface light transport techniques, but create tools that are efficient and enjoyable to use for novice and expert users alike. However, as research works often raise at least as many new questions as they answer, there are, of course, many ways to continue and improve our work.

6.2 Directions For Future Work

We are delighted to see the first works building on these ideas and endeavored to present them in a general manner to be useful for applications in a variety of areas within (and perhaps beyond) computer graphics. In this section, we discuss a selection of already existing follow-up works and point to a few promising directions for further improving these techniques.

6.2.1 Gaussian Material Synthesis

Within the first two years of its appearance, this work has attracted a direct follow-up paper improving our neural renderer [KSSN19], an independent, third-party implementation for the Blender modeling program, and a handful of citations. Our teaser image has also appeared as the cover photo for the Transactions on Graphics journal. We hope that this work will continue to provide fertile ground for potential follow-up works, and can still be extended in a variety of different directions – we briefly summarize the ones we have found to be the most promising ideas. Incorporating active learning schemes [KGUD07] may further improve the sample-efficiency of the initial learning process, thereby requiring fewer interactions from the artist before the recommendations are created. Our multi-round recommendation scheme can also be refined, e.g., by observing the confidence values of different regions in a 2D latent space and tuning the number of per-round samples required to proceed, thereby improving the sample-efficiency of the GPR step. These confidence values can also be used to guide the active learning process towards unexplored regions in the high-dimensional shader space. Moreover, the resolution of the neural renderer depends on the used architecture and the on-board memory of the GPU, and hence, can be improved through AutoML [FKE+15]/architecture search methods [ZL16] and advances in hardware design. Furthermore, the artist needn't be presented with stationary images – these gallery samples can be changed to small animations with different lighting setups or potentially include motion [MLMG19] to further enhance the perception of these materials. We also look forward to experimenting with not only recommending point samples from the latent space, but perform walks that minimize a set of functionals that are useful for material synthesis. We have used uniform sampling to build the training data for this neural network. The accuracy of the neural renderer can likely be further improved through an adaptive, non-uniform sampling of the parameter space for the training set.

6.2.2 Photorealistic Material Editing Through Direct Image Manipulation

This method is capable of inferring a photorealistic material when given an image of a marked up image from the artist. This process was demonstrated on a principled shader model that contains the most commonly used materials (e.g., diffuse, specular, transparent and translucent materials), however, the generality of our formulation offers the possibility of plugging in other kinds of material models. We have named a few promising candidates, e.g., thin-film interference [Dia91, IWR+15], fluorescence [WTP01] birefringence [WW08], microfacet models [HHdD16] layered materials [Bel18, ZJ18], and more. The relative importance of the different image regions can also be emphasized via a weighting function, thereby guiding the optimizer to prioritize a select set of aspects of the drawn input image, such as a specular highlight. Similarly to goal-based caustic design methods [PJJ+11], setting up a scene with a receiver object may also open up the possibility of drawing a desired caustic pattern and letting our two-stage method find a material that casts a caustic pattern of the sought shape.

6.2.3 Separable Subsurface Scattering

This technique was developed as a collaboration with Activision-Blizzard, and has been used in some of their real-time computer game titles. Creating a work that stands its ground in industry applications and also has a solid scientific and mathematical background was quite a challenge. In order to accommodate both sides, with the artist-friendly model, we endeavored to offer a method that maximizes artistic freedom, while our pre-integrated model has proven mathematical guarantees, the SVD-based method provides an easy way to obtain an acceptable quality result within a prescribed (and likely short) time limit. We encourage future research works that strike different tradeoffs that are relevant both for the research community and the industry. We also welcome a variety of creative follow-up works that include the effect of subsurface scattering when creating high-resolution facial scans [FJA$^+$14], build a biophysical model to simulate and render skin aging [IGAJG15], use it for lighting design [SPB16], or even move beyond normal incidence for diffusion profiles and include directionality to the subsurface-scattering process [DCFMB17].

We look forward to seeing these ideas applied to new and unexplored problems within (and beyond) computer graphics and machine learning.

6.3 Acknowledgements, Credit Assignment

As this thesis is aimed at material synthesis and subsurface light transport, a handful of well-made scenes with fine geometry are required to accentuate the material models generated with our methods. In this section, we credit the amazing artistic work of people who created these scenes, followed by a thank you message to everyone who helped us on our journey.

6.3.1 Gaussian Material Synthesis

We would like to thank Robin Marin for the material test scene and Vlad Miller for his help with geometry modeling, Felícia Zsolnai–Fehér for improving the design of many figures, Hiroyuki Sakai, Christian Freude, Johannes Unterguggenberger, Pranav Shyam and Minh Dang for their useful comments, and Silvana Podaras for her help with a previous version of this work. We also thank NVIDIA for providing the GPU used to train our neural networks. This work was partially funded by Austrian Science Fund (FWF), project number P27974. Scene and geometry credits: Gold Bars – JohnsonMartin, Christmas Ornaments – oenvoyage, Banana – sgamusse, Bowl – metalix, Grapes – PickleJones, Glass Fruits – BobReed64, Ice cream – b2przemo, Vases – Technausea, Break Time – Jay–Artist, Wrecking Ball – floydkids, Italian Still Life – aXel, Microplanet – marekv, Microplanet vegetation – macio.

6.3.2 Photorealistic Material Editing Through Direct Image Manipulation

We would like to thank Reynante Martinez for providing us the geometry and some of the materials for the Paradigm (Fig. 4.1) and Genesis scenes (Fig. 4.2), ianofshields for the Liquify scene that served as a basis for Fig. 4.9, 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 NVIDIA for providing the GPU used to train our neural networks. This work was partially funded by Austrian Science Fund (FWF), project number P27974.

6.3.3 Separable Subsurface Scattering

The authors thank Infinite Realities, in particular Lee Perry-Smith, for his head model and for the *Lauren* model; the Institute of Creative Technologies at USC, in particular Paul Debevec, for the *Ari* and *Bernardo* models; and Bernardo Antoniazzi for letting us use his likeness. Furthermore, we want to thank the Stanford University Computer Graphics Laboratory for the *Dragon* model, and the following contributors from Blend Swap under CC-BY licence: longrender for the *Dish* model, metalix for the *Green apple* model, betomo16 for the *Plant* model, and PickleJones for the *Grapes* model. We also thank Felícia Fehér for editing the figures. This research has been partially funded by the European Commission, 7th Framework Programme, through projects GOLEM and VERVE, the Spanish Ministry of Economy and Competitiveness through project LIGHTSLICE, and project TAMA, and the Austrian Science Fund (FWF) through project no. P23700-N23.

List of Figures

List of Tables

List of Algorithms

Bibliography

[AAL16] Miika Aittala, Timo Aila, and Jaakko Lehtinen. Reflectance modeling by neural texture synthesis. *ACM Transactions on Graphics*, 35(4):65, 2016.

[AKW+18] Lynton Ardizzone, Jakob Kruse, Sebastian Wirkert, Daniel Rahner, Eric W Pellegrini, Ralf S Klessen, Lena Maier-Hein, Carsten Rother, and Ullrich Köthe. Analyzing inverse problems with invertible neural networks. *arXiv preprint arXiv:1808.04730*, 2018.

[AKZM14] Melinos Averkiou, Vladimir G Kim, Youyi Zheng, and Niloy J Mitra. Shapesynth: Parameterizing model collections for coupled shape exploration and synthesis. In *Computer Graphics Forum*, volume 33, pages 125–134. Wiley Online Library, 2014.

[AL11] Ken Anjyo and JP Lewis. Rbf interpolation and gaussian process regression through an rkhs formulation. *Journal of Math-for-Industry*, 3(6):63–71, 2011.

[AWL13] Miika Aittala, Tim Weyrich, and Jaakko Lehtinen. Practical svbrdf capture in the frequency domain. *ACM Transactions on Graphics*, 32(4):110–1, 2013.

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

[BAOR06] Aner Ben-Artzi, Ryan Overbeck, and Ravi Ramamoorthi. Real-time brdf editing in complex lighting. In *ACM Transactions on Graphics*, volume 25, pages 945–954. ACM, 2006.

[Bel18] Laurent Belcour. Efficient Rendering of Layered Materials using an Atomic Decomposition with Statistical Operators. *ACM Transactions on Graphics*, 37(4):1, 2018.

[BJLPS17] Piotr Bojanowski, Armand Joulin, David Lopez-Paz, and Arthur Szlam. Optimizing the latent space of generative networks. *arXiv preprint arXiv:1707.05776*, 2017.

[BL03] G. Borshukov and J. P. Lewis. Realistic human face rendering for 'The Matrix Reloaded'. In *ACM SIGGRAPH 2003 Sketches & Applications*, 2003.

[BLNZ95] Richard H Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

[BN03] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

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

[BP09] Bernhard Burgstaller and Friedrich Pillichshammer. The average distance between two points. *Bulletin of the Australian Mathematical Society*, 80(3):353, 2009.

[BR13] Manuel Blum and Martin A Riedmiller. Optimization of gaussian process hyperparameters using rprop. In *ESANN*, pages 339–344, 2013.

[BS12] Brent Burley and Walt Disney Animation Studios. Physically-based shading at disney. In *ACM SIGGRAPH*, volume 2012, pages 1–7, 2012.

[BUSB13] Sean Bell, Paul Upchurch, Noah Snavely, and Kavita Bala. Opensurfaces: A richly annotated catalog of surface appearance. *ACM Transactions on Graphics*, 32(4):111, 2013.

[BVM$^+$17] Steve Bako, Thijs Vogels, Brian Mcwilliams, Mark Meyer, Jan NováK, Alex Harvill, Pradeep Sen, Tony Derose, and Fabrice Rousselle. Kernel-predicting convolutional networks for denoising monte carlo renderings. *ACM Transactions on Graphics*, 36(4):97, 2017.

[BZVL17] Irwan Bello, Barret Zoph, Vijay Vasudevan, and Quoc V Le. Neural optimizer search with reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 459–468. JMLR. org, 2017.

[Cha60] S. Chandrasekhar. *Radiative Transfer*. Dover Books on Intermediate and Advanced Mathematics. Dover Publications, 1960.

[CK14] Neill DF Campbell and Jan Kautz. Learning a manifold of fonts. *ACM Transactions on Graphics*, 33(4):91, 2014.

[CPWAP08] Ewen Cheslack-Postava, Rui Wang, Oskar Akerlund, and Fabio Pellacini. Fast, realistic lighting and material design using nonlinear cut approximation. In *ACM Transactions on Graphics*, volume 27, page 128. ACM, 2008.

[CUH15] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

[CXY$^+$15] Kang Chen, Kun Xu, Yizhou Yu, Tian-Yi Wang, and Shi-Min Hu. Magic decorator: automatic material suggestion for indoor digital scenes. *ACM Transactions on Graphics*, 34(6):232, 2015.

[DAD$^+$18] Valentin Deschaintre, Miika Aittala, Fredo Durand, George Drettakis, and Adrien Bousseau. Single-image svbrdf capture with a rendering-aware deep network. *ACM Transactions on Graphics (TOG)*, 37(4):128, 2018.

[DAD$^+$19] Valentin Deschaintre, Miika Aittala, Fredo Durand, George Drettakis, and Adrien Bousseau. Flexible svbrdf capture with a multi-image deep network. In *Computer Graphics Forum*, volume 38, pages 1–13. Wiley Online Library, 2019.

[DCFMB17] Alessandro Dal Corso, Jeppe Revall Frisvad, Jesper Mosegaard, and J Andreas Baerentzen. Interactive directional subsurface scattering and transport of emergent light. *The Visual Computer*, 33(3):371–383, 2017.

[DI11] Eugene D'Eon and Geoffrey Irving. A quantized-diffusion model for rendering translucent materials. *ACM Transactions on Graphics*, 30(4), 2011.

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

[DJ05] C. Donner and H. W. Jensen. Light diffusion in multi-layered translucent materials. *ACM Transactions on Graphics*, 24(3):1032–1039, 2005.

[DJ07] Craig Donner and Henrik Wann Jensen. Rendering translucent materials using photon diffusion. In *Proceedings of the Eurographics Symposium on Rendering*, pages 243–252, 2007.

[DJ18] Jonathan Dupuy and Wenzel Jakob. An adaptive parameterization for efficient material acquisition and rendering. *Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, December 2018.

[dL07] E. d'Eon and D. Luebke. Advanced techniques for realistic real-time skin rendering. In Hubert Nguyen, editor, *GPU Gems 3*, chapter 14, pages 293–347. Addison Wesley, 2007.

[dLE07] E. d'Eon, D. Luebke, and E. Enderton. Efficient rendering of human skin. In *Proceedings of Eurographics Symposium on Rendering*, pages 147–157, 2007.

[DLR$^+$09] Craig Donner, Jason Lawrence, Ravi Ramamoorthi, Toshiya Hachisuka, Henrik Wann Jensen, and Shree Nayar. An empirical bssrdf model. *ACM Trans. Graph.*, 28(3), 2009.

[EMH18] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *arXiv preprint arXiv:1808.05377*, 2018.

[EY36] Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

[FHK15] Jeppe Revall Frisvad, Toshiya Hachisuka, and Thomas Kim Kjeldsen. Directional dipole model for subsurface scattering. *ACM Trans. Graph.*, To Appear, 2015.

[FJA$^+$14] Graham Fyffe, Andrew Jones, Oleg Alexander, Ryosuke Ichikari, and Paul Debevec. Driving high-resolution facial scans with video performance capture. *ACM Transactions on Graphics (TOG)*, 34(1):8, 2014.

[FKE$^+$15] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. Efficient and robust automated machine learning. In *Advances in neural information processing systems*, pages 2962–2970, 2015.

[Fre15] Christian Freude. Extending separable subsurface scattering to arbitrary materials. Master's thesis, Institute of Computer Graphics and Algorithms, Vienna University of Technology, Favoritenstrasse 9-11/186, A-1040 Vienna, Austria, January 2015.

[GB10] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010.

[GLA$^+$19] Michaël Gharbi, Tzu-Mao Li, Miika Aittala, Jaakko Lehtinen, and Frédo Durand. Sample-based monte carlo denoising using a kernel-splatting network. *ACM Transactions on Graphics (TOG)*, 38(4):125, 2019.

[GLD$^+$19] Duan Gao, Xiao Li, Yue Dong, Pieter Peers, Kun Xu, and Xin Tong. Deep inverse rendering for high-resolution svbrdf estimation from an arbitrary number of images. *ACM Transactions on Graphics (TOG)*, 38(4):134, 2019.

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

[HBH09] John Hable, George Borshukov, and Jim Hejl. Fast skin shading. In Wolfgang Engel, editor, *Shader X7*, chapter 2.4, pages 161–173. Charles River Media, 2009.

[HBR^{+}11] Jing Huang, Tamy Boubekeur, Tobias Ritschel, Matthias Holländer, and Elmar Eisemann. Separable approximation of ambient occlusion. In *Eurographics 2011-Short papers*, 2011.

[HCJ13] Ralf Habel, Per H. Christensen, and Wojciech Jarosz. Photon beam diffusion: A hybrid monte carlo method for subsurface scattering. *Computer Graphics Forum (Proceedings of EGSR 2013)*, 32(4), 2013.

[HHdD16] Eric Heitz, Johannes Hanika, Eugene d'Eon, and Carsten Dachsbacher. Multiple-scattering microfacet bsdfs with the smith model. *ACM Transactions on Graphics (TOG)*, 35(4):58, 2016.

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

[HS52] Magnus Rudolph Hestenes and Eduard Stiefel. *Methods of conjugate gradients for solving linear systems*, volume 49. NBS, 1952.

[HS06] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[HS11] He He and Wan-Chi Siu. Single image super-resolution using gaussian process regression. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 449–456. IEEE, 2011.

[HSF18] Geoffrey E Hinton, Sara Sabour, and Nicholas Frosst. Matrix capsules with em routing. 2018.

[IGAJG15] Jose A Iglesias-Guitian, Carlos Aliaga, Adrian Jarabo, and Diego Gutierrez. A biophysically-based model of the optical properties of skin aging. In *Computer Graphics Forum*, volume 34, pages 45–55. Wiley Online Library, 2015.

[IWR^{+}15] Sho Ikeda, Shin Watanabe, Bisser Raytchev, Toru Tamaki, and Kazufumi Kaneda. Spectral rendering of interference phenomena caused by multilayer films under global illumination environment. *ITE Transactions on Media Technology and Applications*, 3(1):76–84, 2015.

[JB02] H. W. Jensen and J. Buhler. A rapid hierarchical rendering technique for translucent materials. *ACM Transactions on Graphics*, 21(3):576–581, 2002.

[JG10] Jorge Jimenez and Diego Gutierrez. *GPU Pro: Advanced Rendering Techniques*, chapter Screen-Space Subsurface Scattering, pages 335–351. AK Peters Ltd., 2010.

[JJG12] Jorge Jimenez, Adrian Jarabo, and Diego Gutierrez. Separable subsurface scattering. Technical report, Universidad de Zaragoza, 2012.

[JJG15] Adrian Jarabo Christian Freude Thomas Auzinger Xian-Chun Wu Javier von der Pahlen Michael Wimmer Jorge Jimenez, Károly Zsolnai and Diego Gutierrez. Separable subsurface scattering. *Computer Graphics Forum*, 34(6):188–197, 2015.

[JMLH01] H.W. Jensen, S.R. Marschner, M. Levoy, and P. Hanrahan. A practical model for subsurface light transport. In *Proceedings of ACM SIGGRAPH 2001*, pages 511–518, 2001.

[JSG09] J. Jimenez, V. Sundstedt, and D. Gutierrez. Screen-space perceptual rendering of human skin. *ACM Transactions on Applied Perception*, 6(4):1–15, 2009.

[JWSG10] Jorge Jimenez, David Whelan, Veronica Sundstedt, and Diego Gutierrez. Real-time realistic skin translucency. *IEEE Computer Graphics and Applications*, 30(4):32–41, 2010.

[Kaj86] James T Kajiya. The rendering equation. In *ACM SIGGRAPH computer graphics*, volume 20, pages 143–150. ACM, 1986.

[KB14] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[KBS15] Nima Khademi Kalantari, Steve Bako, and Pradeep Sen. A machine learning approach for filtering monte carlo noise. *ACM Transactions on Graphics*, 34(4):122–1, 2015.

[KD13] Anton S Kaplanyan and Carsten Dachsbacher. Path space regularization for holistic and robust light transport. In *Computer Graphics Forum*, volume 32, pages 63–72. Wiley Online Library, 2013.

[KF12] Christopher Kulla and Marcos Fajardo. Importance sampling techniques for path tracing in participating media. *Comp. Graph. Forum*, 31(4):1519–1528, June 2012.

[KGUD07] Ashish Kapoor, Kristen Grauman, Raquel Urtasun, and Trevor Darrell. Active learning with gaussian processes for object categorization. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[KKCF13] Alan King, Christopher Kulla, Alejandro Conty, and Marcos Fajardo. Bssrdf importance sampling. In *ACM SIGGRAPH 2013 Talks*, 2013.

[KM99] Jan Kautz and Michael D McCool. Interactive rendering with arbitrary brdfs using separable approximations. In *Rendering Techniques 99*, pages 247–260. Springer, 1999.

[KMM$^+$17] Simon Kallweit, Thomas Müller, Brian McWilliams, Markus Gross, and Jan Novák. Deep scattering: Rendering atmospheric clouds with radiance-predicting neural networks. *ACM Transactions on Graphics (Proc. of Siggraph Asia)*, 36(6), November 2017.

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

[KSI14] Yuki Koyama, Daisuke Sakamoto, and Takeo Igarashi. Crowd-powered parameter analysis for visual design exploration. In *Proceedings of the 27th annual ACM symposium on User Interface Software and Technology*, pages 65–74. ACM, 2014.

[KSSN19] Aakash KT, Parikshit Sakurikar, Saurabh Saini, and PJ Narayanan. A flexible neural renderer for material visualization. *arXiv preprint arXiv:1908.09530*, 2019.

[KTS$^+$14] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[Law04] Neil D Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In *Advances in Neural Information Processing Systems*, pages 329–336, 2004.

[LBBH98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[LFTG97] Eric PF Lafortune, Sing-Choong Foo, Kenneth E Torrance, and Donald P Greenberg. Non-linear approximation of reflectance functions. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, pages 117–126. ACM Press/Addison-Wesley Publishing Co., 1997.

[LMH$^+$18] Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila. Noise2noise: Learning image restoration without clean data. *arXiv preprint arXiv:1803.04189*, 2018.

[LMS$^+$19] Manuel Lagunas, Sandra Malpica, Ana Serrano, Elena Garces, Diego Gutierrez, and Belen Masia. A similarity measure for material appearance. *arXiv preprint arXiv:1905.01562*, 2019.

[LRR04] Jason Lawrence, Szymon Rusinkiewicz, and Ravi Ramamoorthi. Efficient brdf importance sampling using a factored representation. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 496–505. ACM, 2004.

[LS08] Joel Lehman and Kenneth O Stanley. Exploiting open-endedness to solve problems through the search for novelty. In *ALIFE*, pages 329–336, 2008.

[LS11] Joel Lehman and Kenneth O Stanley. Evolving a diversity of virtual creatures through novelty search and local competition. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, pages 211–218. ACM, 2011.

[LSR$^+$13] Dongping Li, Xin Sun, Zhong Ren, Stephen Lin, Yiying Tong, Baining Guo, and Kun Zhou. Transcut: Interactive rendering of translucent cutouts. *Visualization and Computer Graphics, IEEE Transactions on*, 19(3):484–494, 2013.

[LWC$^+$13] Yiming Liu, Jue Wang, Sunghyun Cho, Adam Finkelstein, and Szymon Rusinkiewicz. A no-reference metric for evaluating the quality of motion deblurring. *ACM Trans. Graph.*, 32(6):175–1, 2013.

[MAB$^+$97] Joe Marks, Brad Andalman, Paul A Beardsley, William Freeman, Sarah Gibson, Jessica Hodgins, Thomas Kang, Brian Mirtich, Hanspeter Pfister, Wheeler Ruml, et al. Design galleries: A general approach to setting parameters for computer graphics and animation. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 389–400. ACM Press/Addison-Wesley Publishing Co., 1997.

[Mac96] David JC MacKay. Bayesian methods for backpropagation networks. In *Models of Neural Networks III*, pages 211–254. Springer, 1996.

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

[MEM19] Gary Mataev, Michael Elad, and Peyman Milanfar. Deepred: Deep image prior powered by red, 2019.

[MES$^+$11] Adolfo Munoz, Jose I. Echevarria, Francisco Seron, Jorge Lopez-Moreno, Mashhuda Glencross, and Diego Gutierrez. BSSRDF estimation from single images. *Computer Graphics Forum*, 30:455–464, 2011.

[MG98] Stephen Robert Marschner and Donald P Greenberg. *Inverse rendering for computer graphics*. Citeseer, 1998.

[MH08] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[Mik10] Morten Mikkelsen. Skin rendering by pseudo-separable cross bilateral filtering. Technical report, Naughty Dog Inc., August 2010.

[MIWI16] Leo Miyashita, Kota Ishihara, Yoshihiro Watanabe, and Masatoshi Ishikawa. Zoematrope: A system for physical material design. In *ACM SIGGRAPH 2016 Emerging Technologies*, page 24. ACM, 2016.

[MKB^{+}05] T. Mertens, J. Kautz, P. Bekaert, F. V. Reeth, and H. P. Seidel. Efficient rendering of local subsurface scattering. *Computer Graphics Forum*, 24(1):41–50, 2005.

[MLMG19] Ruiquan Mao, Manuel Lagunas, Belen Masia, and Diego Gutierrez. The effect of motion on the perception of material appearance. In *ACM Symposium on Applied Perception 2019*, page 16. ACM, 2019.

[MMCS11] Jonathan Masci, Ueli Meier, Dan Cireşan, and Jürgen Schmidhuber. Stacked convolutional auto-encoders for hierarchical feature extraction. *Artificial Neural Networks and Machine Learning–ICANN 2011*, pages 52–59, 2011.

[MMP99] AM Mathai, P Moschopoulos, and G Pederzoli. Random points associated with rectangles. *Rendiconti del Circolo Matematico di Palermo*, 48(1):163–190, 1999.

[Møl93] Martin Fodslette Møller. A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks*, 6(4):525–533, 1993.

[MPBM03] Wojciech Matusik, Hanspeter Pfister, Matt Brand, and Leonard McMillan. A data-driven reflectance model. *ACM Transactions on Graphics*, 22(3):759–769, July 2003.

[MSY16] Xiaojiao Mao, Chunhua Shen, and Yu-Bin Yang. Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In *Advances in neural information processing systems*, pages 2802–2810, 2016.

[NAM^{+}17a] Oliver Nalbach, Elena Arabadzhiyska, Dushyant Mehta, H-P Seidel, and Tobias Ritschel. Deep shading: Convolutional neural networks for screen space shading. In *Computer Graphics Forum*, volume 36, pages 65–78. Wiley Online Library, 2017.

[NAM^{+}17b] Oliver Nalbach, Elena Arabadzhiyska, Dushyant Mehta, Hans-Peter Seidel, and Tobias Ritschel. Deep shading: Convolutional neural networks for screen-space shading. 36(4), 2017.

[Nea12] Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

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

[NHH15] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1520–1528, 2015.

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

[ODO16] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016.

[OSJ$^+$18] Chris Olah, Arvind Satyanarayan, Ian Johnson, Shan Carter, Ludwig Schubert, Katherine Ye, and Alexander Mordvintsev. The building blocks of interpretability. *Distill*, 2018. https://distill.pub/2018/building-blocks.

[PB11] Eric Penner and George Borshukov. *GPU Pro 2*, chapter Pre-Integrated Skin Shading, pages 41–55. AK Peters Ltd., 2011.

[PdMJ14] Marios Papas, Krystle de Mesa, and Henrik Wann Jensen. A physically-based bsdf for modeling the appearance of paper. In *Computer Graphics Forum*, volume 33, pages 133–142. Wiley Online Library, 2014.

[Per14] Emil Persson. Low-level shader optimization for next-gen and DX11. In *Game Developers Conference*, 2014.

[PH04] Matt Pharr and Greg Humphreys. *Physically Based Rendering: From Theory to Implementation*, chapter 8.4.2, pages 382–386. Morgan Kaufmann, 2004.

[PJJ$^+$11] Marios Papas, Wojciech Jarosz, Wenzel Jakob, Szymon Rusinkiewicz, Wojciech Matusik, and Tim Weyrich. Goal-based caustics. In *Computer Graphics Forum*, volume 30, pages 503–511. Wiley Online Library, 2011.

[PRJ$^+$13] Marios Papas, Christian Regg, Wojciech Jarosz, Bernd Bickel, Philip Jackson, Wojciech Matusik, Steve Marschner, and Markus Gross. Fabricating translucent materials using continuous pigment mixtures. *ACM Transactions on Graphics (TOG)*, 32(4):146, 2013.

[PvBM$^+$06] Pieter Peers, Karl vom Berge, Wojciech Matusik, Ravi Ramamoorthi, Jason Lawrence, Szymon Rusinkiewicz, and Philip Dutré. A compact factored representation of heterogeneous subsurface scattering. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 746–753. ACM, 2006.

[RB92] Martin Riedmiller and Heinrich Braun. Rprop-a fast adaptive learning algorithm. In *Proc. of ISCIS VII), Universitat*. Citeseer, 1992.

[RH01] Ravi Ramamoorthi and Pat Hanrahan. A signal-processing framework for inverse rendering. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 117–128. ACM, 2001.

[RJGW19] Gilles Rainer, Wenzel Jakob, Abhijeet Ghosh, and Tim Weyrich. Neural btf compression and interpolation. *Computer Graphics Forum (Proceedings of Eurographics)*, 38(2), March 2019.

[RM51] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[RMS⁺17] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc Le, and Alex Kurakin. Large-scale evolution of image classifiers. *arXiv preprint arXiv:1703.01041*, 2017.

[RSB⁺02] Brigitte Röder, Oliver Stock, Siegfried Bien, Helen Neville, and Frank Rösler. Speech processing activates visual cortex in congenitally blind humans. *European Journal of Neuroscience*, 16(5):930–936, 2002.

[RWG⁺13] Peiran Ren, Jiaping Wang, Minmin Gong, Stephen Lin, Xin Tong, and Baining Guo. Global illumination with radiance regression functions. *ACM Transactions on Graphics*, 32(4):130, 2013.

[SBRCD17] Pỳnar Satỳlmỳs, Thomas Bashford-Rogers, Alan Chalmers, and Kurt Debattista. A machine-learning-driven sky model. *IEEE Computer Graphics and Applications*, 37(1):80–91, 2017.

[SD12] Pradeep Sen and Soheil Darabi. On filtering the noise from the random parameters in monte carlo rendering. *ACM Transactions on Graphics*, 31(3):18–1, 2012.

[SFH17] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In *Advances in Neural Information Processing Systems*, pages 3856–3866, 2017.

[SGM⁺16] Ana Serrano, Diego Gutierrez, Karol Myszkowski, Hans-Peter Seidel, and Belen Masia. An intuitive control space for material appearance. *ACM Transactions on Graphics*, 35(6):186, 2016.

[SHK⁺14] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[SJR18] Tiancheng Sun, Henrik Wann Jensen, and Ravi Ramamoorthi. Connecting measured brdfs to analytic brdfs by data-driven diffuse-specular separation. In *SIGGRAPH Asia 2018 Technical Papers*, page 273. ACM, 2018.

[SKP09] Musawir A. Shah, Jaakko Konttinen, and Sumanta Pattanaik. Image-space subsurface scattering for interactive rendering of deformable translucent objects. *IEEE Computer Graphics and Applications*, 29:66–78, 2009.

[SLD17] Maria Shugrina, Jingwan Lu, and Stephen Diverdi. Playful palette: an interactive parametric color mixer for artists. *ACM Transactions on Graphics*, 36(4):61, 2017.

[SMDH13] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013.

[SNM$^+$13] Thorsten-Walther Schmidt, Jan Novak, Johannes Meng, Anton S. Kaplanyan, Tim Reiner, Derek Nowrouzezahrai, and Carsten Dachsbacher. Path-space manipulation of physically-based light transport. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2013)*, 32(4), aug 2013.

[SPB16] Davoud Shahlaei, Marcel Piotraschke, and Volker Blanz. Lighting design for portraits with a virtual light stage. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 1579–1583. IEEE, 2016.

[SSN18] Cyril Soler, Kartic Subr, and Derek Nowrouzezahrai. A versatile parameterization for measured material manifolds. In *Computer Graphics Forum*, volume 37, pages 135–144. Wiley Online Library, 2018.

[Sta95] Jos Stam. Multiple scattering as a diffusion process. In *Eurographics Rendering Workshop*, pages 41–50, 1995.

[STPP09] Ying Song, Xin Tong, Fabio Pellacini, and Pieter Peers. Subedit: a representation for editing measured heterogeneous subsurface scattering. *ACM Transactions on Graphics (TOG)*, 28(3):31, 2009.

[SWH$^+$16] Shunsuke Saito, Lingyu Wei, Liwen Hu, Koki Nagano, and Hao Li. Photorealistic facial texture inference using deep neural networks. *arXiv preprint arXiv:1612.00523*, 2016.

[SZC$^+$07] Xin Sun, Kun Zhou, Yanyun Chen, Stephen Lin, Jiaoying Shi, and Baining Guo. Interactive relighting with dynamic brdfs. *ACM Transactions on Graphics*, 26(3):27, 2007.

[TDSL00] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[TKL$^+$16] Apostolia Tsirikoglou, Joel Kronander, Per Larsson, Tanaboon Tongbuasirilai, Andrew Gardner, and Jonas Unger. Differential appearance editing for measured brdfs. In *ACM SIGGRAPH 2016 Talks*, page 51. ACM, 2016.

[VLL$^+$10] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(Dec):3371–3408, 2010.

[WBS$^+$04] Zhou Wang, Alan C Bovik, Hamid R Sheikh, Eero P Simoncelli, et al. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[WD97] David J Wales and Jonathan PK Doye. 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):5111–5116, 1997.

[WFH08] Jack M Wang, David J Fleet, and Aaron Hertzmann. Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):283–298, 2008.

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

[WJZ95] Lihong Wang, Steven L Jacques, and Liqiong Zheng. ”monte carlo modeling of light transport in multi-layered tissues. *Computer methods and programs in biomedicine*, 47(2):131–146, 1995.

[WTP01] Alexander Wilkie, Robert F Tobler, and Werner Purgathofer. Combined rendering of polarization and fluorescence effects. In *Rendering Techniques 2001*, pages 197–204. Springer, 2001.

[WW08] Andrea Weidlich and Alexander Wilkie. Realistic rendering of birefringency in uniaxial crystals. *ACM Transactions on Graphics (TOG)*, 27(1):6, 2008.

[WWH$^+$10] Yajun Wang, Jiaping Wang, Nicolas Holzschuch, Kartic Subr, Jun-Hai Yong, and Baining Guo. Real-time rendering of heterogeneous translucent objects with arbitrary shapes. In *Computer Graphics Forum*, volume 29, pages 497–506. Wiley Online Library, 2010.

[WZT$^+$08] Jiaping Wang, Shuang Zhao, Xin Tong, Stephen Lin, Zhouchen Lin, Yue Dong, Baining Guo, and Heung-Yeung Shum. Modeling and rendering of heterogeneous translucent materials using the diffusion equation. *ACM Trans. Graph.*, 27(1):9:1–9:18, March 2008.

[ZFWW18] Károly Zsolnai-Fehér, Peter Wonka, and Michael Wimmer. Gaussian material synthesis. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 2018.

[ZFWW19] Károly Zsolnai-Fehér, Peter Wonka, and Michael Wimmer. Photorealistic material editing through direct image manipulation, 2019.

[ZH05] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.

[ZJ18] Tizian Zeltner and Wenzel Jakob. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 37(4):74:1–74:14, July 2018.

[ZKSE16] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *European Conference on Computer Vision*, pages 597–613. Springer, 2016.

[ZKTF10] Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Rob Fergus. Deconvolutional networks. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2528–2535. IEEE, 2010.

[ZL16] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

[ZM09] Xiang Zhu and Peyman Milanfar. A no-reference sharpness metric sensitive to blur and noise. In *2009 International Workshop on Quality of Multimedia Experience*, pages 64–69. IEEE, 2009.