RESEARCH-ARTICLE

Practical Inverse Rendering of Textured and Translucent Appearance

PHILIPPE WEIER, Saarland University, Saarbrucken, Saarland, Germany

JÉRÉMY RIVIERE, Google Switzerland GmbH, Zurich, ZH, Switzerland

RUSLAN GUSEINOV, Google LLC, Europe, Dublin, Ireland

STEPHAN JOACHIM GARBIN, Google LLC, Europe, Dublin, Ireland

PHILIPP SLUSALLEK, Saarland University, Saarbrucken, Saarland, Germany

BERND BICKEL, Google Switzerland GmbH, Zurich, ZH, Switzerland

View all

Open Access Support provided by:

Google LLC, Europe

Google Switzerland GmbH

Saarland University

Published: 01 August 2025
Accepted: 29 March 2025
Received: 22 January 2025

Practical Inverse Rendering of Textured and Translucent Appearance

PHILIPPE WEIER, Google and Saarland University, Germany
JÉRÉMY RIVIERE, Google, Switzerland
RUSLAN GUSEINOV, Google, Austria
STEPHAN GARBIN, Google, United Kingdom
PHILIPP SLUSALLEK, Saarland University, Germany
BERND BICKEL, Google, Switzerland
THABO BEELER, Google, Switzerland
DELIO VICINI, Google, Switzerland

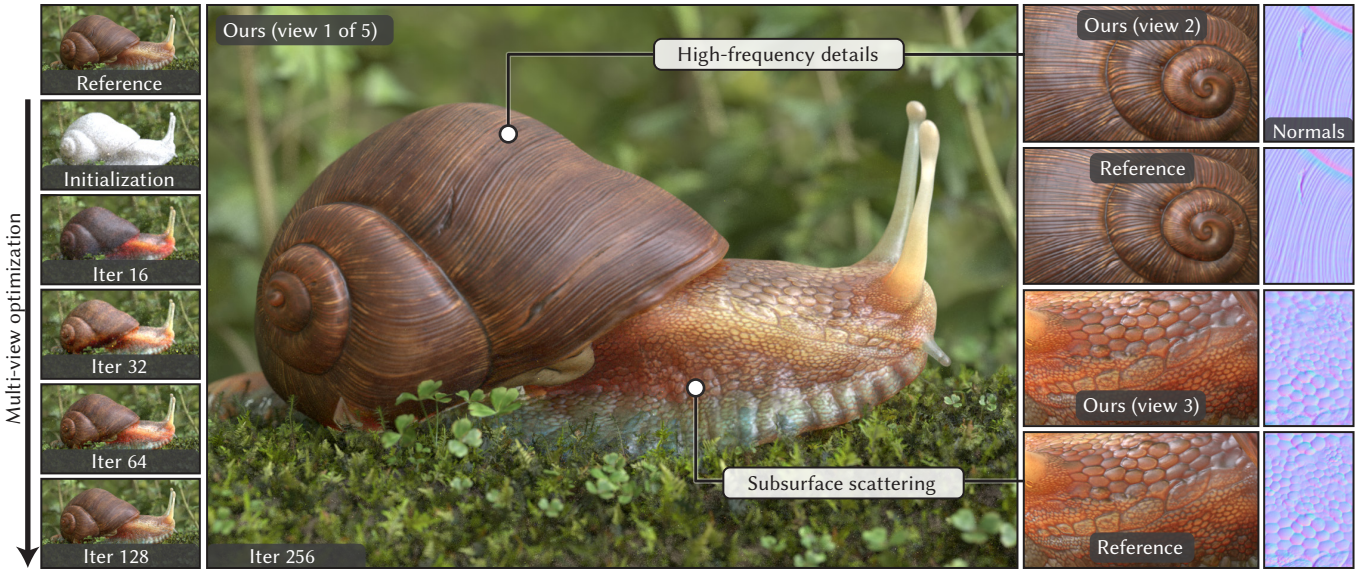

Fig. 1. Inverse reconstruction of high-frequency normal maps and subsurface scattering (SSS) parameters on a synthetic scene. **Left column:** Starting from known geometry and lighting, we reconstruct appearance textures of the snail using differentiable path tracing. We fit normal maps, SSS albedo maps as well as uniform, spectrally-varying extinction and phase function parameters from synthetic multi-view data consisting of five camera views and eight lighting conditions (obtained by rotating the ground-truth environment lighting). **Center column:** Our method reconstructs appearance textures that allow high-quality re-rendering. **Right column:** We show insets comparing both rendering and recovered normal maps to the corresponding ground truth.

Inverse rendering has emerged as a standard tool to reconstruct the parameters of appearance models from images (e.g., textured BSDFs). In this work, we present several novel contributions motivated by the practical challenges of recovering high-resolution surface appearance textures, including spatially-varying subsurface scattering parameters.

First, we propose *Laplacian mipmapping*, which combines differentiable mipmapping and a Laplacian pyramid representation into an effective preconditioner. This seemingly simple technique significantly improves the quality of recovered surface textures on a set of challenging inverse rendering problems. Our method automatically adapts to the render and texture resolutions, only incurs moderate computational cost and achieves better quality than prior work while using fewer hyperparameters. Second, we introduce a specialized gradient computation algorithm for textured, path-traced subsurface scattering, which facilitates faithful reconstruction of translucent materials. By using path tracing, we enable the recovery of complex appearance while avoiding the approximations of the previously used diffusion dipole methods. Third, we demonstrate the application of both these techniques to reconstructing the textured appearance of human faces from sparse captures. Our method recovers high-quality relightable appearance parameters that are compatible with current production renderers.

Authors' Contact Information: Philippe Weier, weier@cg.uni-saarland.de, Google and Saarland University, Germany; Jérémy Riviere, jmriviere@google.com, Google, Switzerland; Ruslan Guseinov, ruslanguseinov@google.com, Google, Austria; Stephan Garbin, stephangarbin@google.com, Google, United Kingdom; Philipp Slusallek, slusallek@cg.uni-saarland.de, Saarland University, Germany; Bernd Bickel, berndbickel@google.com, Google, Switzerland; Thabo Beeler, tbeeler@google.com, Google, Switzerland; Delio Vicini, vicini@google.com, Google, Switzerland.

ACM 1557-7368/2025/8-ART117
https://doi.org/10.1145/3730855

CCS Concepts: • **Computing methodologies → Rendering**.

Additional Key Words and Phrases: inverse rendering, differentiable rendering, Laplacian pyramid, mip mapping, subsurface scattering, skin appearance

ACM Reference Format:
Philippe Weier, Jérémy Riviere, Ruslan Guseinov, Stephan Garbin, Philipp Slusallek, Bernd Bickel, Thabo Beeler, and Delio Vicini. 2025. Practical Inverse Rendering of Textured and Translucent Appearance. *ACM Trans. Graph.* 44, 4, Article 117 (August 2025), 16 pages. https://doi.org/10.1145/3730855

1 Introduction

Physically-based inverse rendering methods differentiate light transport simulations to minimize image-based objective functions over the space of scene parameters. These methods have become a standard tool for 3D geometry and appearance reconstruction. The work in this paper is motivated by the practical challenges encountered when using inverse rendering to reconstruct textures. Our goal is to recover high-quality appearance textures (e.g., albedo, roughness and normal maps) of physically-based material models from images of a scene (e.g., photographs).

A first challenge is that reconstructing textures using standard gradient descent often suffers from uneven convergence, noisy results and undesirable local minima. A primary reason for this is the highly heterogeneous nature of inverse rendering problems. For example, an albedo texture and a normal map affect the objective function differently. Moreover, different views of a scene might observe textured details at different scales. Both rendered images and gradients are further corrupted by Monte Carlo noise, with the noise level varying between different views or lighting conditions. These properties of inverse rendering make it hard to choose the right hyperparameters for existing preconditioning [Nicolet et al. 2021] or gradient filtering [Chang et al. 2024] methods. We introduce a novel preconditioning technique that significantly improves reconstructed textures by combining Laplacian image pyramids [Burt and Adelson 1983] and mipmapping [Williams 1983]. Our *Laplacian mipmapping* is easy to implement, has low computational overhead and significantly outperforms prior work on a number of inverse rendering problems. It is less sensitive to hyperparameters than previous approaches, and therefore easier to apply to complex scenes. We later validate this by showing results for a scene with over 100 textures and 30 different materials.

A second practical challenge is that many real objects are translucent and exhibit subsurface scattering effects. Frequently, the subsurface scattering is heterogeneous and varies throughout the object. In production rendering, it is common to approximate this heterogeneity using a 2D texture, which sets the scattering parameters of a given subsurface light path from its entry point on the object's surface [Burley et al. 2018]. This offers a good proxy for the internal variation, while avoiding costly 3D textures. We introduce a variant of *path replay backpropagation* [Vicini et al. 2021b] that efficiently differentiates path-traced subsurface scattering with textured parameters. Using path tracing avoids the limitations of previously used diffusion dipole models [Deng et al. 2022; Jensen et al. 2001]. We show that our method, in conjunction with our Laplacian mipmapping, robustly recovers textured subsurface scattering parameters. Figure 1 shows the results of a complex joint optimization of normal maps and spectrally-varying subsurface scattering.

Finally, we apply both our Laplacian mipmapping and differentiable subsurface scattering to the challenging problem of facial appearance reconstruction. Creating realistic digital humans is a long-standing area of research, with applications ranging from visual effects to synthetic data generation for machine learning. Our methods, for the first time, enable using end-to-end differentiable path tracing to directly optimize the parameters of standard production rendering models for human skin. The reconstructed appearance textures enable realistic rendering with novel views and lighting conditions. A key insight is that existing physically-based rendering models can achieve a high degree of realism if their parameters are optimized using inverse rendering.

The remainder of the paper is organized around our three core contributions, which are:

- Laplacian mipmapping, improving robustness and quality of inverse rendering of textures.
- Efficient differentiation of textured, path-traced subsurface scattering, facilitating inverse rendering of spatially- and spectrally-varying translucent materials.
- An application of the previous two techniques to a challenging dataset of sparse multi-view captures of human faces.

2 Related work

Our review of related work mirrors the structure of the rest of the paper. We first discuss work related to texture reconstruction and prefiltering, then continue with subsurface scattering and conclude with a short summary of facial appearance capture.

2.1 Texture reconstruction and prefiltering

Physically-based inverse rendering. Prior work on physically-based inverse rendering primarily focused on obtaining accurate gradients of geometric visibility discontinuities [Bangaru et al. 2020; Li et al. 2018; Loubet et al. 2019; Zhang et al. 2020] and global illumination [Nimier-David et al. 2020; Vicini et al. 2021b], as well as providing the underlying rendering systems [Jakob et al. 2022b,a]. Our goal is to use the resulting gradients to reconstruct high-quality textures. Nicolet et al. [2021] address a similar problem for inverse rendering of meshes. Instead of directly optimizing an array of 3D vertex positions, they optimize in *differential coordinates* [Sorkine 2006]. This *Laplacian preconditioner* significantly improves the reconstructed meshes and can also be applied to textures. Recently, Chang et al. [2024] used a cross-bilateral filter [Eisemann and Durand 2004; Petschnigg et al. 2004] to denoise texture gradients, which can further improve results in some settings. We compare our Laplacian mipmapping to both of these approaches, and show consistent improvements while requiring less hyperparameter tuning.

Radiance field methods. Multiscale representations and mipmapping are commonly used to improve robustness and quality of *neural radiance fields (NeRFs)* [Mildenhall et al. 2020]. Barron et al. [2021] suggest prefiltering the NeRF's positional features based on ray differentials. On the other hand, neural hash grids [Müller et al. 2022] accumulate features across a hash grids of increasing resolutions, and have also been combined with mipmapped lookups [Barron et al. 2023]. Accumulating features across hash grid resolutions is conceptually similar to our use of a Laplacian pyramid. Several works have

used some form of pyramidal texture representation [Turki et al. 2023; Verbin et al. 2024]. PyNerf [Turki et al. 2023] combines a pyramid of grid-based neural features with mipmapped lookups. Our Laplacian mipmapping takes inspiration from these prior works, but does not require neural networks. This decreases the evaluation cost and makes our representation suitable for physically-based inverse rendering.

Prefiltering for rendering. Mipmapping [Williams 1983] is a type of *prefiltering* that reduces noise and aliasing artifacts caused by mismatching texture and render resolution. By tracing ray differentials [Igehy 1999], a renderer can select the texture of appropriate resolution from a precomputed hierarchy of downsampled textures. Conventional linear downsampling works well for color textures, but cannot always preserve rendered appearance [Bruneton and Neyret 2012]. Non-linear texture prefiltering methods improve results for normal mapped [Olano and Baker 2010] and displacement mapped [Dupuy et al. 2013] surfaces. The combination of these methods with our work is an interesting future direction. Finally, neural material models [Kuznetsov et al. 2021; Xue et al. 2024] commonly build on multiscale representations to support mipmapped lookups, and volumetric representations [Bako et al. 2023; Loubet and Neyret 2017; Vicini et al. 2021a; Weier et al. 2023; Zhou et al. 2025] have been used to prefilter complex geometry.

2.2 Subsurface scattering

Forward rendering. Rendering subsurface scattering requires simulating volumetric scattering to solve the *radiative transfer equation (RTE)* [Chandrasekhar 1960], e.g., using path tracing [Kajiya 1986; Kajiya and Von Herzen 1984]. Early works in computer graphics used *diffusion theory* [d'Eon et al. 2007; Donner and Jensen 2005; Jensen et al. 2001], which approximates the solution to the RTE using analytical scattering profiles. While fast to compute, these methods produce visible rendering bias on thin geometry or high curvature regions. Modern production renderers have thus largely abandoned diffusion in favor of path-traced subsurface scattering [Burley et al. 2018; Christensen et al. 2018; Georgiev et al. 2018]. The path tracing estimator can be made more efficient by using Dwivedi sampling [Křivánek and d'Eon 2014], path guiding [Herholz et al. 2019] or specialized light sampling techniques [Koerner et al. 2016]. We show how to adapt path-traced subsurface scattering to inverse rendering problems. An alternative approach are neural subsurface scattering estimators [Leonard et al. 2021; Tg et al. 2024; Vicini et al. 2019], which allow some rendering error in favor of performance.

Inverse rendering. Inverse rendering of volumetric scattering has applications such as the reconstruction of general participating media [Gkioulekas et al. 2016, 2013], fabric appearance [Khungurn et al. 2015] and prefiltering of volumetric appearance [Zhao et al. 2016]. Velinov et al. [2018] used path-traced subsurface scattering to model the appearance of teeth. Many early works were limited to computing derivatives of a small number of parameters, whereas our implementation does not impose any restriction on the number of optimized parameters. Recently, Deng et al. [2022] differentiated the diffusion approximation [Jensen et al. 2001] to reconstruct textured subsurface scattering. Our method produces more realistic results due to the use of the accurate path tracing estimator.

2.3 Facial appearance capture

Appearance capture. Accurately reconstructing the appearance of human faces is an important problem in computer graphics. Over the years, a variety of light stage capture systems have been proposed [Debevec et al. 2000; Ghosh et al. 2011; Guo et al. 2019; Ma et al. 2007]. These typically feature a large number of cameras and some combination of active illumination, structured light and polarized imaging. Early works often imposed lengthy, multistage capture protocols. For example, Weyrich et al. [2006] measured subsurface scattering using purpose-built contact probes that had to be placed on the skin. Ghosh et al. [2008] projected dot patterns to recover subsurface scattering profiles, in addition to using polarization to recover fine details. See also Klehm et al. [2015] for a survey covering early works. Later approaches reduced complexity and cost of capture setups [Alexander et al. 2010; Fyffe et al. 2017; Riviere et al. 2020] and leveraged biophysical parameterizations [Aliaga et al. 2022]. Most closely related to our work are inverse rendering methods [Gotardo et al. 2018; Ha et al. 2024; Riviere et al. 2020]. Specifically, Riviere et al. [2020] use diffusion-based texture-space subsurface scattering with fixed scattering parameters and Ha et al. [2024] optimize biophysical parameters of a two-layer diffusion model. However, neither of these methods use full differentiable path tracing. Our approach closes the gap between the algorithms used at capture time and in a production renderer.

Neural rendering. The challenges of realistic rendering of faces have given rise to a large body of work on neural rendering and representations for faces [Bai et al. 2023; Lombardi et al. 2018; Rao et al. 2022; Sarkar et al. 2023; Sun et al. 2021; Xu et al. 2023; Zhang et al. 2021]. While these methods achieve a high degree of realism, they are largely incompatible with physically-based rendering and the resulting representations cannot be used in a standard production renderer.

3 Laplacian mipmapping

3.1 Motivation

Inverse rendering. Our goal is to reconstruct textures that minimize an image-based objective function. Formally, we use gradient descent to solve:

$$\hat{\boldsymbol{\pi}} = \arg\min_{\boldsymbol{\pi}} g(I(\boldsymbol{\pi})), \tag{1}$$

where $\boldsymbol{\pi}$ denotes a vector of optimized parameters (e.g., a texture), I is the rendering function and g is a differentiable objective function (e.g., L_2 difference to a target image).

Desired properties. The space of possible solutions to improve texture reconstruction is large. We design our method to satisfy the following key properties, which we found relevant for practical applications:

(1) **Minimal hyperparameter tuning:** We want to avoid introducing additional hyperparameters that need to be tuned per texture map, as tuning such parameters quickly becomes impractical for complex scenes.

(2) **Resolution independent:** Related to the above, we want texture optimization to be as independent of render and texture resolution as possible. Modifying either of those should not dramatically alter our method's optimal hyperparameters.
(3) **Computationally efficient:** Most applications of inverse rendering are to some extent limited by computational cost. Thus, changes to the texture representation should incur minimal additional computations. Texture optimization needs to scale to very high resolutions (e.g., 8192^2 texels).
(4) **Hardware agnostic:** While GPUs are common for gradient-based optimizations, high-end CPUs can achieve good performance for inverse rendering [Jakob et al. 2022a]. Texture optimization should work on both GPU and CPU, and not require hardware acceleration of specific operations (e.g., matrix multiplication).

Limitations of existing approaches. Before introducing our method, we briefly discuss the limitations of existing approaches to texture optimization. A straightforward solution is using gradient descent with the *Adam optimizer* [Kingma and Ba 2015], without any special handling of textures. In practice, this often requires a large number of iterations to converge and is prone to noise artifacts in the reconstructed textures and local minima. One possible remedy are *coarse-to-fine* schemes. Starting from a low initial resolution, the optimized texture is upsampled at fixed steps (e.g., every N iterations). Finding the right upsampling "schedule" is difficult, and depends on the type of texture, rendering resolution and amount of Monte Carlo noise. The upsampling step decouples fitting of low and high frequencies, and unconverged low-resolution textures impede fitting of fine details later on. Another possible improvement is using *differentiable mipmapping* [Williams 1983]. For this, one first computes a Gaussian pyramid [Adelson et al. 1984] of the texture by repeated differentiable downsampling. The differentiable renderer then accesses the appropriate texture resolution using the current ray's differential [Igehy 1999]. From all mipmap levels, gradients "propagate" up to the high-resolution base texture. This reduces artifacts for scenes with significant texture *minification*, i.e., when the texel density exceeds the rendered pixel density. Otherwise, this performs similarly to standard Adam and does not accelerate fitting of low frequencies. We will separately discuss and compare to prior work by Nicolet et al. [2021] and Chang et al. [2024] in Section 3.3.

3.2 Method

Laplacian mipmapping. Our approach is rendering-resolution-aware, automatically adapts to the texture's frequency content and does not require any scheduling mechanism or additional hyperparameters. We leverage the well-known duality between Laplacian and Gaussian pyramids. While Gaussian pyramids store downsampled images, Laplacian pyramids store residuals between the lower resolutions and the next higher resolution. Each level of a Gaussian pyramid (i.e., a mipmapped texture) can be efficiently constructed from its corresponding Laplacian pyramid. Instead of the mipmapped texture itself, we optimize the levels of the Laplacian pyramid, which act as an (overparameterized) preconditioner for the optimized texture.

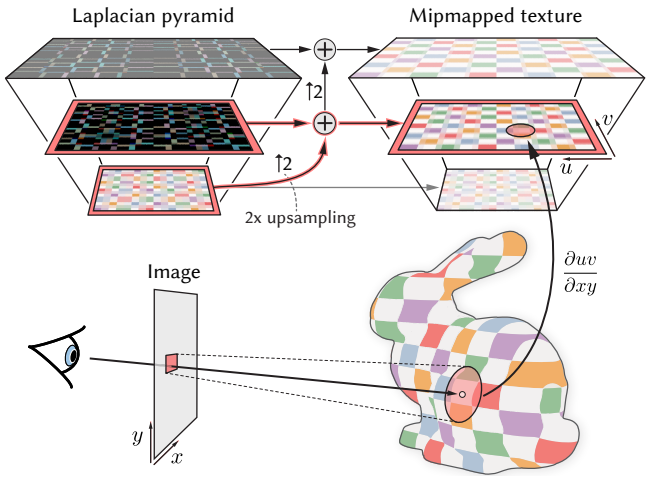

Fig. 2. We construct a texture's mipmap hierarchy from a Laplacian pyramid. A texture lookup during rendering thus accumulates information from all levels in the Laplacian pyramid *below* the queried mipmap level (red).

Figure 2 illustrates our approach and highlights the levels of the Laplacian pyramid contributing when a specific mipmap level is accessed. Conventional differentiable mipmapping propagates gradients from lower resolutions to the original high-resolution texture, whereas our Laplacian mipmapping averages and downsamples gradients to lower-resolution pyramid levels. Our approach combines the benefits of coarse-to-fine schemes and mipmapping, while avoiding their shortcomings. Most importantly, it significantly reduces the resolution dependence of the optimization's convergence, making it easier to handle complex and heterogeneous inverse rendering problems.

Clamping and frequency decomposition. Many appearance parameters require restriction to a valid parameter range during optimization (e.g., albedo values have to be in $[0, 1]$). We support this by clamping the highest-resolution mipmap level to the desired range and rebuilding the Laplacian pyramid. This has the side effect of ensuring frequency decomposition between levels. Therefore, a finer resolution cannot contain low frequencies already accounted for at the lower levels. We found this to slightly improve results, although it can be a trade-off in the presence of non-linear minification effects, as we will discuss in the following section.

3.3 Results and discussion

Implementation details. We implemented our method using Dr.Jit and Mitsuba [Jakob et al. 2022b,a]. Timings are measured on a system with an AMD Ryzen Threadripper PRO 3945WX 12-core CPU and an NVIDIA RTX A6000 GPU. Unless stated otherwise, optimizations run for 128 gradient descent steps. We compare our method to previous work by Nicolet et al. [2021] and Chang et al. [2024]. Nicolet et al. [2021] provide a Cholesky solver for their linear system, but this is impractical for high-resolution textures (e.g., the Cholesky decomposition for a 4096^2 texture takes 13 minutes). We therefore implemented a conjugate gradient solver using Dr.Jit, which is

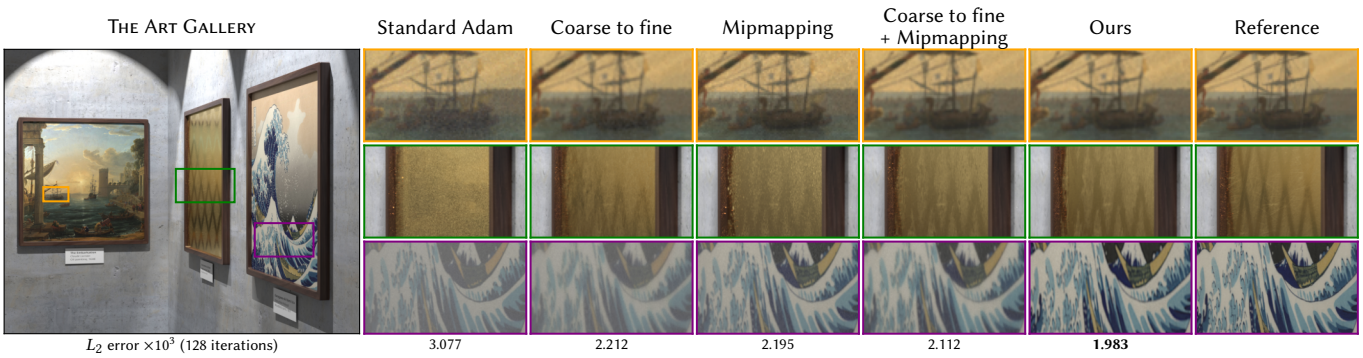

Fig. 3. Comparison of Laplacian mipmapping to existing approaches. We compare our method to standard Adam, coarse-to-fine texture upsampling, differentiable mipmapping and the combination of coarse to fine and mipmapping. From left to right, we optimize each painting's texture of the following type and resolution: albedo (2048^2), roughness (4096^2) and albedo (8192^2). The rendering resolution is 1500×1200.

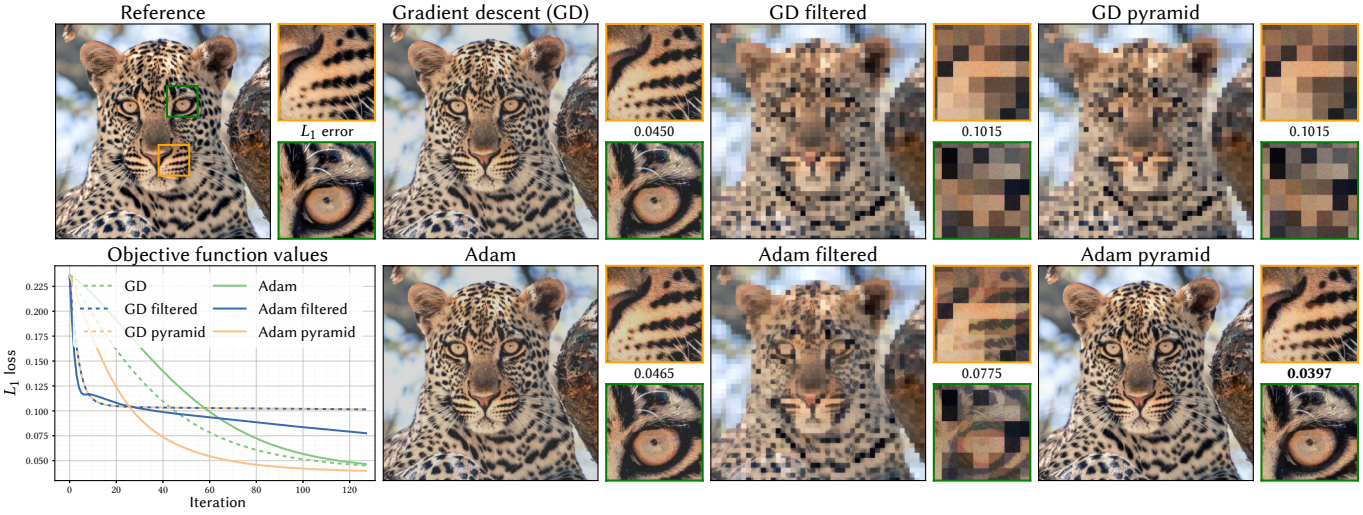

Fig. 4. Optimization of an image that is "rendered" by querying a texture on a regular grid of the same size. **Top row:** Unmodified gradient descent on a two-level pyramid is equivalent to linearly filtering the gradients of a regular non-pyramid texture. **Bottom row:** In contrast, the Adam optimizer improves results for the pyramid by adjusting the learning rate per texel *per level*. For completeness, we also provide results for unmodified gradient descent and Adam.

specialized to regular texture grids and scales better to higher resolutions. For the cross-bilateral gradient filtering, we follow Chang et al. [2024] and implement it using À-trous filters [Dammertz et al. 2010]. Our Dr.Jit implementation matches the original Slang [He et al. 2018] code and achieves similar performance.

Comparison to coarse to fine & mipmapping. In Figure 3, we compare our method to existing approaches on a challenging single-view optimization. For each method, we determine the learning rate that minimizes the final L_2 loss using a brute-force search. Coarse to fine with an exponential upsampling schedule improves over standard Adam, but cannot converge uniformly across all textures. Due to the mismatch between texture and render resolution, mipmapping improves convergence and high-frequency details. Combining coarse to fine with mipmapping further decreases the error, but is overly smooth due to insufficient iterations at the finest resolution. Finally, our Laplacian mipmapping achieves uniform convergence for all optimized textures, recovering both low- and high-frequency details more consistently than the other methods.

Analysis of pyramid representation. Figure 4 considers a simplified problem, which helps to further analyze the source of these improvements. We optimize a texture that is "rendered" by evaluating it on an image grid of matching resolution. To emulate Monte Carlo rendering noise, we jitter the evaluation of the texture within the image pixels and add a small amount of Gaussian color noise. We use a two-level pyramid with level resolutions 64^2 and 2048^2, and use nearest neighbor upsampling for the lower level (instead of the usual bilinear upsampling). The rendering does not use mipmapping and thus always evaluates the sum of both levels.

This setup reveals an interesting equivalence: Using the pyramid representation is equivalent to optimizing a regular, non-pyramid texture while *filtering* its gradient before applying the gradient descent step. The filter that has to be applied is the sum of the

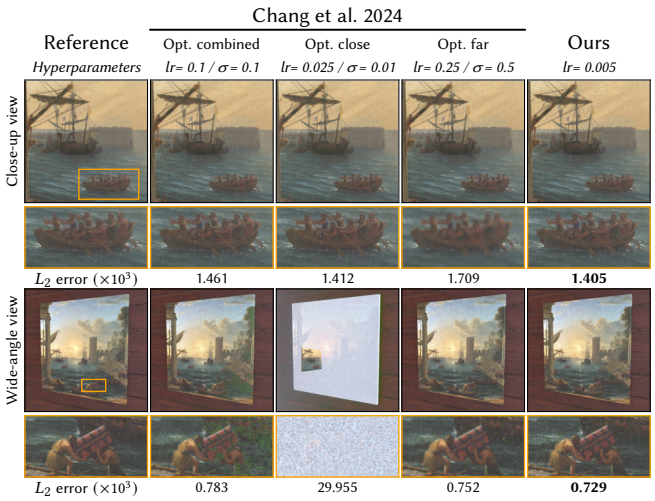

Fig. 5. Comparison of our method to cross-bilateral gradient filtering [Chang et al. 2024] on a scene where a 4096^2 texture is simultaneously optimized for a close-up (top row) and a wide-angle view (bottom row), each rendered at 1024^2. We find the optimal learning rate (lr) and cross-bilateral filter weight σ using a grid search. We do this for both views at the same time ("Opt. combined"), as well as for each view individually ("Opt. close" and "Opt. far"). The optimal parameters for one view do not generalize to the other, and even the best combined filter bandwidth σ cannot simultaneously handle both views.

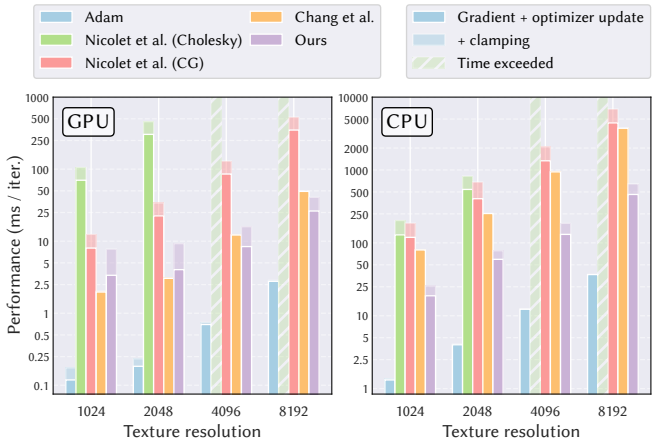

Fig. 6. We plot the time for differentiable forward evaluation, gradient computation, and, if applicable, clamping for our method and prior work over varying texture resolutions. For Nicolet et al. [2021] we show timings for both the original Cholesky solver and our conjugate gradient (CG) solver. The Cholesky solver's preprocessing time is not included in the plot. At 4096^2, it exceeds 10 min and we therefore only report timings up to 2048^2.

identity and a 32×32 box-filtered downsampling followed by nearest-neighbor upsampling, see the supplemental material for details. The equivalence holds for conventional gradient descent. The Adam optimizer, however, adapts the per-level step size and can leverage the pyramid's overparameterization to adjust to the problem at hand. In summary, using an adaptive optimizer such as Adam is essential for the pyramid representation to improve results. We always use the same base learning rate for all pyramid levels, and did not find consistent improvements using any other strategy (e.g., reducing learning rates for lower resolutions).

Hyperparameter tuning challenges. In Figure 5, we compare our approach to the cross-bilateral gradient filtering [Chang et al. 2024] in a two-camera setup. The gradient filtering struggles to simultaneously fit to both views, since the ideal filter bandwidth σ differs between them. A similar limitation affects the smoothing parameter λ in the Laplacian preconditioner [Nicolet et al. 2021], see the supplemental material. Our method successfully handles the variation in texture and render resolution and outperforms both prior methods.

Performance. We evaluate the runtime performance of our method and prior work in Figure 6. We plot the *texture update cost* per optimization step. The reported time consists of the cost of a differentiable forward evaluation of the given representation, followed by a reverse-mode gradient computation and, if applicable, any non-trivial clamping or filtering steps. It does not include render time, as this is highly scene dependent. On GPU, our method performs similar to the cross-bilateral filtering. On CPU, it is up to 4× faster. Not surprisingly, standard Adam is very fast, as it only performs a small amount of computation to update the optimizer state. In practice, the cost for these texture updates is dominated by rendering cost, and can further be amortized by rendering several views in a given iteration. This makes it such that the relative overhead of our method over standard Adam becomes small.

Comparison to previous work. In Figure 7, we evaluate the reconstruction quality of our method against prior works [Chang et al. 2024; Nicolet et al. 2021]. Learning rates and, if applicable, additional hyperparameters are tuned per scene. For a given scene, all textures use the same learning rate, except normal maps, which across techniques benefit from scaling the base learning rate by 0.1. Normal maps require additional care to handle backfacing normals, see the supplemental material for details. The challenging Patio scene has over 30 optimized materials, each using albedo, roughness, metalness and normal maps. The texture resolutions range from 1024^2 to 4096^2. The optimization uses six cameras and the rendering resolution is gradually increased from 240×135 to 1920×1080 using an exponential schedule. Previous work struggles to handle the diverse texture and image resolutions. Our Laplacian mipmapping excels in this scene, providing stable and uniform convergence across all textures, while requiring minimal hyperparameter tuning. Our supplemental HTML viewer allows to interactively explore results for all the tested hyperparameter values (>60 for previous work). In the Leather Bag scene, we optimize a 4096^2 normal map responsible for high-frequency glint-like appearance using a close-up camera view (at 1024^2 resolution) and four different lighting conditions. In this setup, the Laplacian preconditioner outperforms standard Adam and gradient filtering. As also noted by Chang et al. [2024], the cross-bilateral filter overblurs textures without sufficiently distinct structural features, and in this case performs worse than standard Adam. After 32 iterations, our approach already outperforms standard Adam's final result. Finally, the Wicker Basket uses an

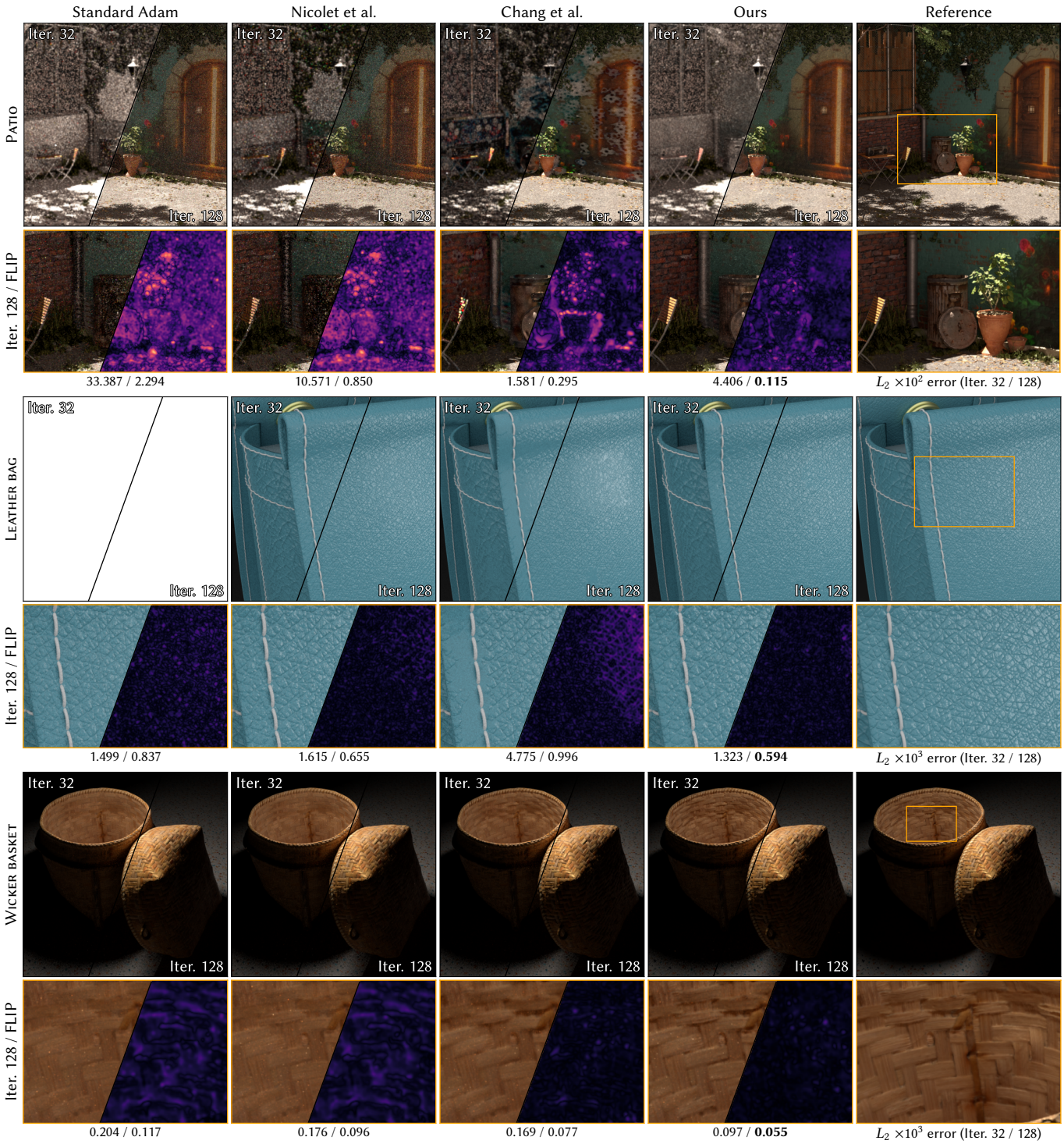

Fig. 7. We compare our method, Laplacian preconditioning [Nicolet et al. 2021] and cross-bilateral gradient filtering [Chang et al. 2024], and visualize FLIP error [Andersson et al. 2020]. We report the L_2 error for the full image after 32 and 128 iterations.

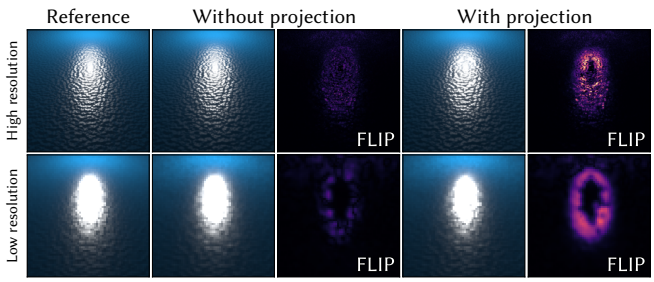

Fig. 8. We render a high-frequency normal map with constant roughness at a 1024^2 and 32^2 render resolution, respectively, and optimize textured roughness (2048^2) and normal map (4096^2). We show results with and without projection (i.e., rebuilding of the Laplacian pyramid) at every iteration.

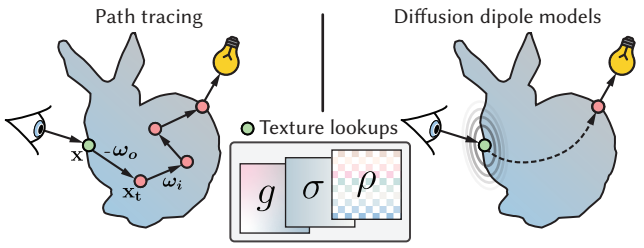

Fig. 9. **Left:** Path-traced subsurface scattering estimates outgoing radiance by sampling a light path through the object's interior. The medium parameters are set by evaluating 2D textures at the point of entry. **Right:** An alternative, approximate solution are diffusion dipole models [Jensen et al. 2001], which directly sample outgoing surface locations according to a two-dimensional scattering profile [King et al. 2013].

identical setup as the previous scene, albeit with an 8196^2 normal map. The cross-bilateral filter is a good fit for the structured appearance of the wicker, but our method still provides a higher quality final reconstruction. Across all scenes, a learning rate of 0.005 or 0.001 consistently worked well for our approach, highlighting its robustness in adapting to texture and rendering resolution.

Limitations. Our clamping rebuilds the Laplacian mipmap at every gradient step, which is a form of projected gradient descent. The example in Figure 8 shows that this might produce worse results when non-linear prefiltering is required. We constructed this optimization setup to highlight the well-known non-linear interplay between roughness and normal map [Olano and Baker 2010]. In this case, linearly filtering when rebuilding the pyramid is incorrect, and we get better results by disabling the projection step. In practice, we still found the projection to generally benefit results on our other scenes. For future work, it would be interesting to combine our method with approaches such as LEAN [Olano and Baker 2010] or LEADR [Dupuy et al. 2013] mapping.

4 Inverse subsurface scattering

4.1 Rendering model

Volume rendering. We now turn to reconstructing materials with textured subsurface scattering. Their translucent appearance is described by the radiative transfer equation [Chandrasekhar 1960], which models the interaction of light with participating media. It can be reformulated as the recursive *volume rendering equation*:

$$L_o(\mathbf{x}, \boldsymbol{\omega}_o) = \int_0^\infty T(\mathbf{x}, \mathbf{x}_t)\sigma\rho \int_{\mathcal{S}^2} f_p(\boldsymbol{\omega}_o, \boldsymbol{\omega}_i) L_i(\mathbf{x}_t, \boldsymbol{\omega}_i)\, \mathrm{d}\boldsymbol{\omega}_i\, \mathrm{d}t, \quad (2)$$

where L_o and L_i are outgoing and incident radiance, $\boldsymbol{\omega}_o$ and $\boldsymbol{\omega}_i$ direction vectors, $\mathbf{x}$ a 3D position, $\mathbf{x}_t \coloneqq \mathbf{x} - \boldsymbol{\omega}_o \cdot t$, and $\mathcal{S}^2$ the unit sphere. The participating medium's appearance is determined by *extinction coefficient* σ, *single-scattering albedo* ρ and *phase function* f_p. For a homogeneous medium, the *transmittance* is defined as $T(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{y} - \mathbf{x}\|\,\sigma)$. See [Pharr et al. 2023] for a comprehensive introduction to volume rendering.

Path-traced subsurface scattering. Path tracing estimates the outgoing radiance due to subsurface scattering by applying Monte Carlo integration to Equation (2). We sample a light path through an object's interior by alternating between sampling the free-flight distance to the next scattering event and the outgoing direction. The distance is sampled proportional to transmittance, and the direction according to the phase function. As common in production rendering [Chiang et al. 2016], we approximate the scattering medium's three-dimensional heterogeneity using a set of 2D surface textures. During rendering, the used medium parameters are determined by the surface position at which the current light path entered the object. Figure 9 illustrates the path-traced subsurface scattering algorithm, and contrasts it to the diffusion approximation used by prior work [Deng et al. 2022; Jensen et al. 2001].

4.2 Inverse rendering

Differentiable path tracing. Differentiating the above estimator using conventional automatic differentiation would require a prohibitive amount of memory to store intermediate computation state. We therefore use *path replay backpropagation (PRB)* [Vicini et al. 2021b], which leverages arithmetic invertibility and recomputation to avoid storing expensive state checkpoints. At a high level, PRB computes gradients in three phases:

(1) A forward, also called *primal*, rendering pass estimates the rendered image and differentiates the objective function with respect to the image.
(2) A second uncorrelated rendering pass traces new light paths and retains their accumulated radiance estimates.
(3) A final pass *replays* the paths from step 2 by using the same random seed. It accumulates parameter gradients by leveraging the previously computed radiance.

Baseline implementation. We implemented a differentiable path tracing integrator that supports subsurface scattering and PRB. We render spectrally-varying media using multiple importance sampling (MIS) [Veach and Guibas 1995] over color channels [Pharr et al. 2023, Section 14.2.3]. Our specialized implementation does not need to handle fully heterogeneous media or next-event estimation within the medium, which results in an up to 4× speedup over Mitsuba's [Jakob et al. 2022b] general-purpose volumetric path tracer with MIS. To support differentiating textured parameters, each iteration of the volumetric path tracing loop needs to differentiably

```
1  def accumulate_subsurface_scattering_gradient(δ_L, L, ...):
2    x_surface = ray_intersect(ray, ...)
3    ...
4    while current path in a scattering medium:
5
6      σ, ρ, ... = evaluate_volume_parameters(x_surface, π, ...)
7      t, T = sample_transmittance(σ, ...)
8      ω_i = sample_phase_function(ray, ...)
9
10     β_i = ρ * σ * T
11     δ_π += backward(δ_L * L * β_i / detach(β_i))
12     ...
```

Algorithm 1. Pseudocode for computing gradients of textured subsurface scattering with path replay backpropagation.

```
1  def accumulate_subsurface_scattering_gradient(δ_L, L, ...):
2    N = 0, t̂ = 0.0
3    x_surface = ray_intersect(ray, ...)
4
5    σ, ρ, ... = evaluate_volume_parameters(x_surface, π, ...)
6    while current path in scattering medium:
7      t, T = sample_transmittance(σ, ...)
8      ω_i = sample_phase_function(ray, ...)
9      t̂ += t
10     N += 1
11     ...
12
13   # Equation (4)
14   δ_π += backward(δ_L * L * (N * ρ * σ / detach(ρ * σ) - t̂ * σ))
15   ...
```

Algorithm 2. Pseudocode for the improved PRB that avoids repeated differentiation of textured parameters.

re-evaluate the medium parameters and backpropagate gradients related to the current iteration.

Pseudocode. Algorithm 1 provides pseudocode for the gradient accumulation, or *adjoint*, third phase of PRB. Following Vicini et al. [2021b]'s notation, δ_L is the derivative of the loss function with respect to radiance. Further, L is the radiance retained from the second phase of PRB and $\mathbf{x}_{\text{surface}}$ is the point of entry. Lines 6–8 perform standard volumetric path sampling and lines 10–11 accumulate parameter derivatives. The function `backward` evaluates gradients using local reverse-mode automatic differentiation, and `detach` detaches a quantity from the gradient computation. Similar to the original PRB, no automatic differentiation over loop boundaries is used, which avoids storing large checkpoints.

Limitations of Algorithm 1. The above implementation is correct, but wasteful: Despite the medium parameters being fully determined by the point of entry, the algorithm repeatedly differentiates the associated bilinear texture fetches. The loop body performs four atomic additions (or eight, if mipmapping is used) to the gradient of each used texture, which is expensive due the high number of interactions required for subsurface scattering.

Improving performance. We reduce the number of atomic additions by aggregating the gradient computation. Omitting the phase function for simplicity, the throughput of a subsurface scattering

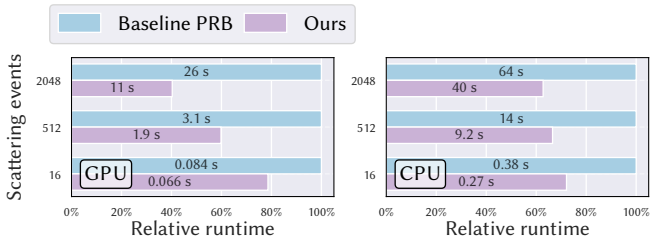

Fig. 10. Performance of our methods when differentiating textured albedo, extinction and phase function (texture resolution of 1024^2, render resolution of 512^2, 1 sample per pixel). The horizontal axis is the relative runtime compared to the baseline PRB, and the numbers in each bar are absolute timings. We render three sets of medium parameters, each corresponding to a different average number of scatterings events along a path.

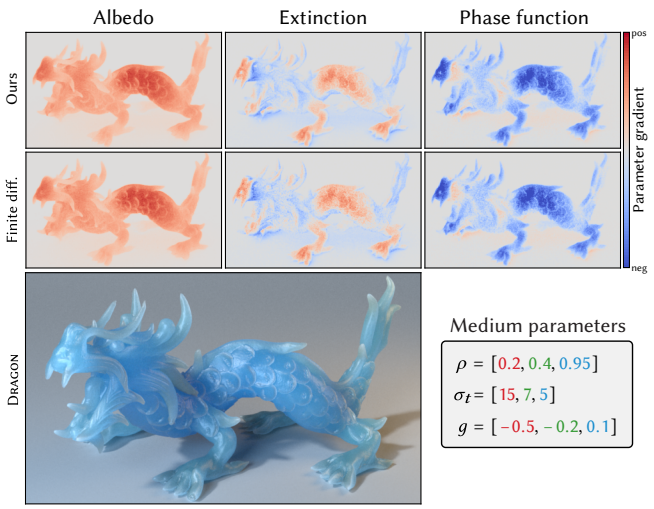

Fig. 11. We validate our implementation's forward-mode gradients for spectrally-varying albedo, extinction and Henyey-Greenstein phase function anisotropy g against finite differences.

light path is:

$$\beta = \prod_{i=1}^{N} \beta_i = \rho^N \sigma^N e^{-\hat{t}\sigma}, \tag{3}$$

where N is the number of scattering events in the volume and $\hat{t}$ the path's length. We use standard PRB for the outer path tracing loop over surface interactions, but aggregate all backward passes involving the volumetric path throughput update $\beta_i/\texttt{detach}(\beta_i)$ (line 11 in Algorithm 1) into a single evaluation of the ratio of $\partial_\pi \beta$ and β. We compute this ratio robustly by simplifying:

$$\frac{\partial_\pi \beta}{\beta} = \frac{\partial_\pi \left[\rho^N \sigma^N e^{-\hat{t}\sigma_t}\right]}{\rho^N \sigma^N e^{-\hat{t}\sigma_t}} = \frac{N \partial_\pi [\rho\sigma]}{\rho\sigma} - \hat{t}\partial_\pi \sigma. \tag{4}$$

This expression can be evaluated using automatic differentiation and is numerically robust for large values of N or $\hat{t}$. Algorithm 2 shows the modified algorithm, which avoids re-evaluating textured parameters and gradient accumulation within the loop. This idea

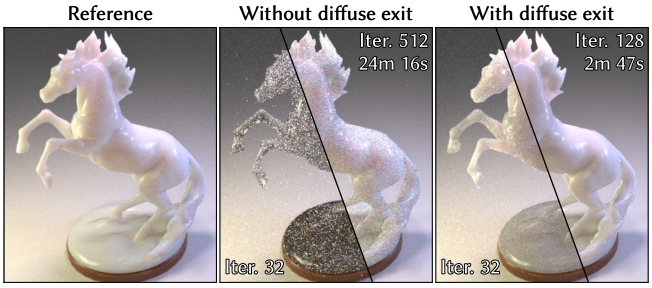

Fig. 12. Impact of using a diffuse exit BSDF on inverse rendering. **Left:** The reference is rendered *without* diffuse exit. **Center:** Unbiased rendering without diffuse exit needs 512 iterations to converge due to the high rendering noise. **Right:** Our optimization with diffuse exit manages to closely approximate the reference within 128 iterations, and leads to lower variance and render times. Both final optimization states are rendered at a higher sample count to better showcase the quality difference. The reported timings are for the entire optimization.

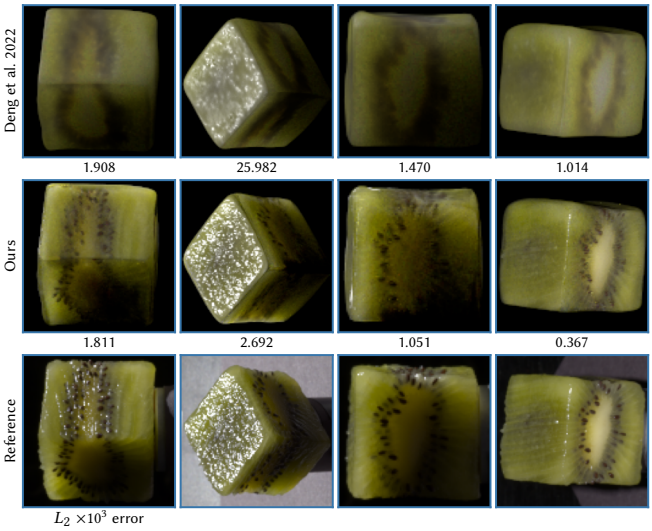

Fig. 13. Comparison of our method and the differentiable diffusion dipole model [Deng et al. 2022].

generalizes to computing phase function gradients, and we provide pseudocode for it in the supplemental material.

Figure 10 shows a performance comparison between both our implementations. The improved version is between 20% and 60% faster, depending on the average number of scattering events. In Figure 11, we verify the correctness of our method by comparing forward-mode derivatives to finite differences[1]. Our estimator is unbiased and closely matches the finite-difference gradients.

Variance reduction. Path-traced subsurface scattering suffers from high variance under non-diffuse illumination, which also affects estimated parameter gradients. We therefore use an approximation that reduces variance to a practical level. As a light path exits the scattering medium, we evaluate a diffuse transmission lobe, instead of the original rough dielectric boundary BSDF. This approach has been used in various production renderers [Chiang et al. 2016]. Figure 12 shows the impact of this approximation on optimization results. It reduces variance and improves convergence, at the cost of a small amount of bias. More sophisticated next-event estimation techniques could be used if unbiased rendering is required [Koerner et al. 2016].

Optimization. We use our Laplacian mipmapping to optimize textured subsurface scattering parameters. We found that directly optimizing the single-scattering albedo ρ does not work well, since its effect on the final image pixels is highly non-linear. For example, a light path with 100 interactions will have a final contribution that is ρ^{100}. A seemingly high albedo of $\rho = 0.99$ will produce an *effective albedo* $\bar{\rho}$ of $\bar{\rho} \approx 0.36$, but $\rho = 0.9995$ reaches an effective albedo of 0.95. We reduce this non-linearity by instead optimizing effective albedo values and differentiably computing single-scattering albedo using the remapping introduced in [Pharr et al. 2018, Section 15.5.8]:

$$\rho = \frac{1 - e^{-8\bar{\rho}}}{1 - e^{-8}}. \tag{5}$$

We found this to consistently lead to better optimization results. In production rendering, similar mappings are used to provide more intuitive controls for manual editing of scattering materials [Chiang et al. 2016; Wrenninge et al. 2017].

4.3 Results

Synthetic scenes. In Figure 14, we show optimization results using our differentiable path tracing together with Laplacian mipmapping. We recover a textured subsurface scattering albedo (2048^2 texture), spectrally-varying extinction (RGB vector), and the Henyey-Greenstein phase function's anisotropy parameter g (RGB vector). Both examples use a single camera view and eight different lighting conditions obtained by rotating the environment map lighting. We render the results under novel illumination, demonstrating that the reconstructed parameters generalize to lighting conditions not used during optimization.

Teaser. In Figure 1, we increase the optimization complexity of the Snail scene by also recovering high-frequency normal maps, in addition to the medium parameters. We use four more camera views and increase the number of optimization steps to 256 to ensure high-quality results. Our supplemental video shows the optimization process on different camera views and lighting conditions, and the supplemental HTML viewer allows detailed inspection of the results.

Captured data. Deng et al. [2022] provide a dataset of real captures of translucent objects. For each object, the dataset contains 50 images with varying lighting conditions. Deng et al. [2022] use a differentiable implementation of a dipole diffusion model to reconstruct the translucent appearance. In Figure 13, we compare our approach to theirs, demonstrating the joint benefit of our Laplacian mipmapping and differentiable path-traced subsurface scattering. Since our method does not handle geometry optimization, we first reconstruct the geometry using their implementation and re-run

[1]Any PRB implementation can be extended to support forward-mode derivatives by replacing the inner reverse-mode gradient computation (`backward`) by the corresponding forward-mode computation. This is useful for validation and debugging, as it allows to render "gradient images".

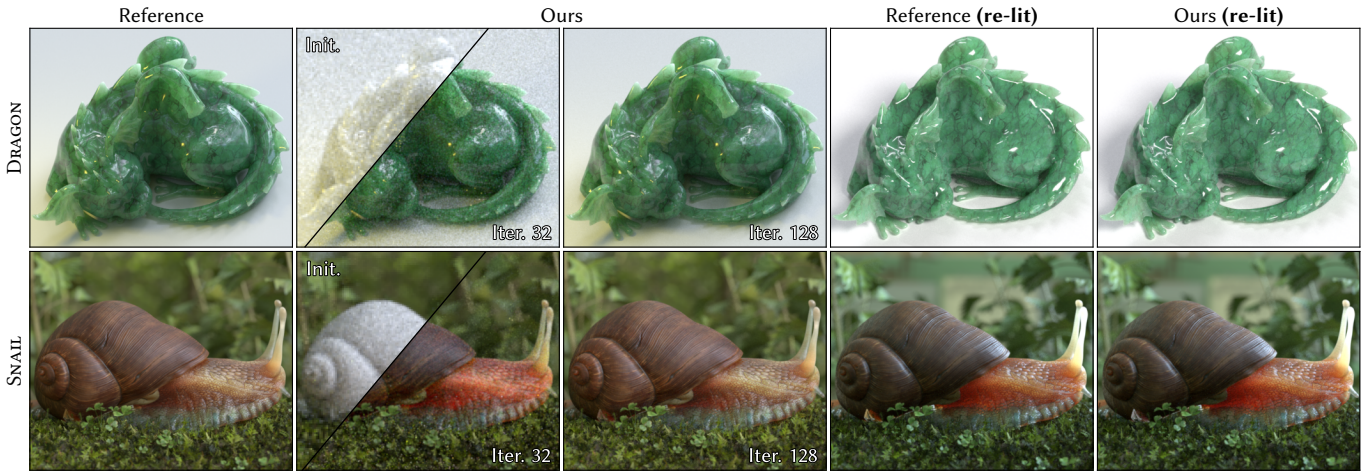

Fig. 14. Inverse rendering of subsurface scattering albedo texture, spectrally-varying extinction and phase-function parameters. On the Snail scene, only the shell and skin of the snail are optimized.

the appearance-only optimization for both methods (see the supplemental material for details). Our method achieves a more faithful reconstruction of the kiwi, capturing intricate translucency effects that are hard (or impossible) for dipole diffusion models to recover.

5 Application: Facial appearance reconstruction

5.1 Problem setup

We use our methods to reconstruct appearance parameters that accurately reproduce the high-frequency specular highlights and subsurface scattering observed on human faces. Our focus is recovering the appearance of skin, and we do not aim to build detailed models of hair or eyes.

Data capture. We use a sparse multi-view capture setup consisting of 13 cameras and 14 lights that are evenly distributed over the hemisphere in front of the captured subject's face, similar to [Sarkar et al. 2023]. Each capture includes both one-light-at-a-time (OLAT) images and interspersed "full-on" frames, where all lights are on.

Preprocessing. The full-on captures are used to perform a multi-view stereo reconstruction of the subject, to which we register a parametric face model [Blanz and Vetter 1999]. This ensures clean base mesh geometry with a consistent UV layout, although our method could also be directly applied to the initial multi-view stereo mesh. Further, we compute optical flow between full-on frames and use the resulting motion vector field to align OLAT frames. This compensates for small motion of the subject during capture [Guo et al. 2019]. We obtain an initial guess of the light source positions by capturing images of a mirror sphere.

5.2 Method

Overview. Given initial geometry, light positions and calibrated camera poses, we use inverse rendering to reconstruct textured appearance parameters from the OLAT images. By leveraging our Laplacian mipmapping and differentiable subsurface scattering, this becomes feasible and requires only little additional complexity. We focus on the high-level ideas here, and provide detailed optimization settings in the supplemental material.

Light representation. We model the physical light sources using a measured *IES profile*, which describes their shape, size and radial falloff, and was obtained from a commercial vendor. The light sources are approximated as disks ($\approx$ 9 cm diameter) with a radially-symmetrical angular falloff, and are rendered using a custom Mitsuba emitter plugin. This representation improves over simpler heuristics such as point or sphere lights. While the importance of accurate lighting representations has been highlighted in previous work [Ling et al. 2024], we believe to be the first to use measured IES profiles for inverse rendering. Finally, we use a smooth falloff of the light intensity towards the edge of the disk to differentiably render shadows without costly discontinuity handling.

Multiresolution optimization. We would like to fit appearance maps to captured images of resolution 2048×3072. However, rendering at this resolution in every optimization step would be extremely costly. We therefore start at a resolution of 16×24 and gradually increase the resolution as the optimization progresses. This reduces total optimization time. Thanks to our Laplacian mipmapping, initially rendering at lower resolutions does not harm the recovery of fine details later on.

Two-stage optimization. To further reduce optimization time, we split the optimization in two phases and only enable subsurface scattering in the second phase. We begin by optimizing albedo and roughness of a principled BSDF [Burley 2012], alongside a high-resolution normal map (4096^2) and a low-resolution displacement map (256^2). While we also recover eye textures, these merely serve as low-quality proxies. We further optimize the light source position, orientation and intensity, as well as a diagonal per-camera color gain. The latter compensates for imperfections in the initial color calibration. We use path replay backpropagation [Vicini et al. 2021b] to differentiably render with up to one indirect light bounce. After 136 iterations, we run a second optimization to fit the subsurface

Fig. 15. Results of our method on captured faces. We compare the re-renderings of our method, both with and without subsurface scattering (SSS), to the captured reference images. Columns 4-6 show zoom-ins of the images in columns 1-3.

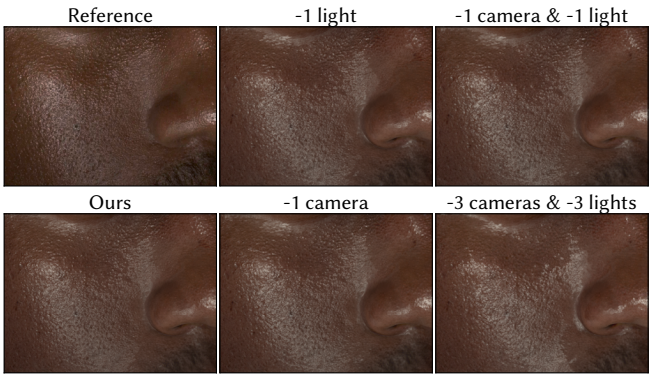

Fig. 16. Effect of leaving out some camera views and lights. We show results (without subsurface scattering) of not using the current camera view or light, or both in the optimization. We also show the effect of holding out two additional cameras and lights.

scattering parameters. For this, we optimize textured subsurface scattering albedo (4096^2), extinction (128^2) and Henyey-Greenstein anisotropy (128^2). The difference in texture resolutions encodes the assumption that density and anisotropy vary at lower frequency than albedo. We keep normal, displacement and roughness map fixed and optimize the subsurface scattering parameters for another 136 iterations. The first optimization phase takes around 4 h 15 min, and the second around 1 h. The second phase is faster since we use only 8 randomly sampled OLAT images in each iteration, in contrast to all 182 in the first phase. We found this to be sufficient, as the subsurface scattering has less angular variation than the specular highlights that we fit in the first phase.

5.3 Results

Appearance reconstruction. Figure 15 shows the output of our method after both first and second optimization phase. The first phase recovers challenging specular highlights and fine surface detail. Without subsurface scattering, the skin looks unrealistic, in particular for light hitting at grazing angles. Adding subsurface scattering smooths out the diffuse appearance, while retaining specular highlights. As expected, subsurface scattering is slightly less pronounced for darker skin tones. Overall, we achieve a high-quality reconstruction using an appearance model that matches current production rendering implementations. See the supplemental video for animations with moving cameras and lights.

Data sparsity. Due to the sparse capture setup, the results in Figures 15, 17 and 18 use all available OLAT images, i.e., the entirety of the captured dataset. In Figure 16, we analyze the effect of leaving out one camera, one light or both from the optimization. We show how the results degrade further when removing two more cameras and lights. The quality visibly decreases as more data is removed. Conversely, we believe that using a dense light stage, such as [Guo et al. 2019], would likely further improve results.

Relighting. The recovered appearance parameters generalize to novel lighting conditions. In Figure 17, we render reconstructions

Fig. 17. Renderings of our reconstructions under environment illumination.

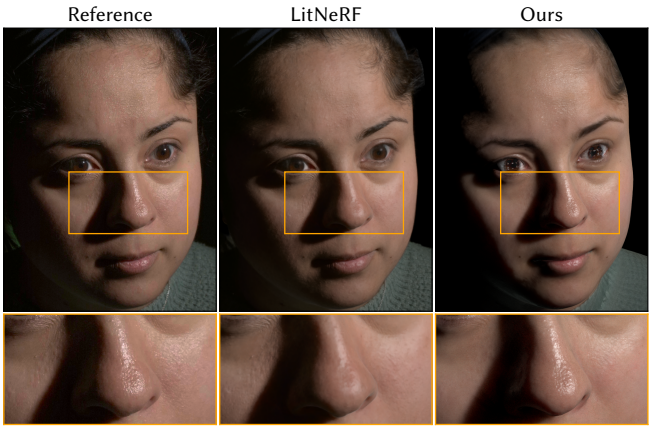

Fig. 18. We compare our method to LitNeRF [Sarkar et al. 2023]. The used view and lighting condition are part of the optimization for both methods.

under environment map lighting, demonstrating the robustness of our appearance parameters and the realism that can be achieved.

Comparison to LitNeRF. Finally, we also compare our method to LitNeRF [Sarkar et al. 2023], a recent NeRF-based representation for OLAT captures that achieves relighting using a neural intrinsic decomposition. Figure 18 shows that the NeRF representation allows for a plausible reconstruction of features such as hair, eyes or clothes, but suffers from significant blurring of specular highlights. Our approach more accurately recovers high-frequency specular reflections. Moreover, the NeRF-based approach does not allow seamless integration into conventional production renderers.

6 Conclusion

Summary. We presented Laplacian mipmapping and differentiable textured subsurface scattering, which together expand the utility of inverse rendering for the recovery of textured materials. We demonstrate this on a challenging dataset of human faces, obtaining high-quality rendering by fitting commonly used production rendering appearance models.

Limitations and future work. Our Laplacian mipmapping currently only supports linear prefiltering and leveraging non-linear prefiltering methods such as LEAN mapping [Olano and Baker 2010] remains future work. Moreover, it would be interesting to combine it with

anisotropic or stochastic texture filtering [Pharr et al. 2024]. For the differentiable subsurface scattering, the main limitation is the performance of the primal rendering itself. This could be addressed by adapting more advanced variance-reduction techniques [Herholz et al. 2019; Koerner et al. 2016; Křivánek and d'Eon 2014]. Finally, the results on facial appearance could be improved by using a denser capture setup or polarization information. We hope that our work can provide a building block to develop future, improved skin appearance models. It would further be interesting to investigate specialized techniques to better reconstruct eyes and hair.

Acknowledgments

We thank Paulo Gotardo for useful discussions, Kripasindhu Sarkar for helping with the LitNeRF comparison, and Cynthia Herrera for supporting data capture. The supplemental HTML viewer is based on code by Alexander Rath. We also thank the various artists for the textures and scenes used in this paper, including Austin Michaud, Giles Laurent, Haron Tosh, Mat Karmon and Rous Kristall. Philippe Weier has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No 956585.

References

Edward H. Adelson, Charles H. Anderson, James R. Bergen, Peter J. Burt, and Joan M. Ogden. 1984. Pyramid methods in image processing. *RCA Eng.* 29, 6 (1984), 33–41.

Oleg Alexander, Mike Rogers, William Lambeth, Jen-Yuan Chiang, Wan-Chun Ma, Chuan-Chang Wang, and Paul Debevec. 2010. The Digital Emily Project: Achieving a Photorealistic Digital Actor. *IEEE Computer Graphics and Applications* 30, 4 (2010), 20–31. doi:10.1109/MCG.2010.65

Carlos Aliaga, Christophe Hery, and Mengqi Xia. 2022. Estimation of spectral biophysical skin properties from captured RGB albedo. *arXiv preprint arXiv:2201.10695* (2022).

Pontus Andersson, Jim Nilsson, Tomas Akenine-Möller, Magnus Oskarsson, Kalle Åström, and Mark D. Fairchild. 2020. ꟻLIP: A Difference Evaluator for Alternating Images. *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 3, 2 (2020), 15:1–15:23. doi:10.1145/3406183

Ziqian Bai, Feitong Tan, Zeng Huang, Kripasindhu Sarkar, Danhang Tang, Di Qiu, Abhimitra Meka, Ruofei Du, Mingsong Dou, Sergio Orts-Escolano, Rohit Pandey, Ping Tan, Thabo Beeler, Sean Fanello, and Yinda Zhang. 2023. Learning Personalized High Quality Volumetric Head Avatars from Monocular RGB Videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Steve Bako, Pradeep Sen, and Anton Kaplanyan. 2023. Deep Appearance Prefiltering. *ACM Trans. Graph.* 42, 2, Article 23 (Jan. 2023). doi:10.1145/3570327

Sai Bangaru, Tzu-Mao Li, and Frédo Durand. 2020. Unbiased Warped-Area Sampling for Differentiable Rendering. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)* 39, 6 (2020), 245:1–245:18. doi:10.1145/3414685.3417833

Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. 2021. Mip-NeRF: A Multiscale Representation for Anti-Aliasing Neural Radiance Fields. In *IEEE International Conference on Computer Vision (ICCV)*.

Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. 2023. Zip-NeRF: Anti-Aliased Grid-Based Neural Radiance Fields. In *IEEE International Conference on Computer Vision (ICCV)*.

Volker Blanz and Thomas Vetter. 1999. A morphable model for the synthesis of 3D faces. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. 187–194.

Eric Bruneton and Fabrice Neyret. 2012. A Survey of Nonlinear Prefiltering Methods for Efficient and Accurate Surface Shading. *IEEE Trans. on Visualization and Computer Graphics* 18, 2 (2012), 242–260. doi:10.1109/TVCG.2011.81

Brent Burley. 2012. Physically-based shading at Disney. *SIGGRAPH Course Notes. Practical physically-based shading in film and game production.* (2012), 1–27.

Brent Burley, David Adler, Matt Jen-Yuan Chiang, Hank Driskill, Ralf Habel, Patrick Kelly, Peter Kutz, Yining Karl Li, and Daniel Teece. 2018. The Design and Evolution of Disney's Hyperion Renderer. *ACM Trans. Graph.* 37, 3 (July 2018), 33:1–33:22. doi:10/gfjm8w

Peter J. Burt and Edward H. Adelson. 1983. The Laplacian Pyramid as a Compact Image Code. *IEEE Trans. on Communications* 31, 4 (1983), 532–540. doi:10.1109/TCOM.1983.1095851

Subrahmanyan Chandrasekhar. 1960. *Radiative transfer*. Dover publications, New York.

Wesley Chang, Xuanda Yang, Yash Belhe, Ravi Ramamoorthi, and Tzu-Mao Li. 2024. Spatiotemporal Bilateral Gradient Filtering for Inverse Rendering. In *SIGGRAPH Asia Conference Papers* (Tokyo, Japan). ACM, 11 pages. doi:10.1145/3680528.3687606

Matt Jen-Yuan Chiang, Peter Kutz, and Brent Burley. 2016. Practical and controllable subsurface scattering for production path tracing. In *SIGGRAPH Talks* (Anaheim, California). Article 49, 2 pages. doi:10.1145/2897839.2927433

Per Christensen, Julian Fong, Jonathan Shade, Wayne Wooten, Brenden Schubert, Andrew Kensler, Stephen Friedman, Charlie Kilpatrick, Cliff Ramshaw, Marc Bannister, Brenton Rayner, Jonathan Brouillat, and Max Liani. 2018. RenderMan: An Advanced Path-Tracing Architecture for Movie Rendering. *ACM Trans. Graph.* 37, 3 (Aug. 2018), 30:1–30:21. doi:10/gfznbs

Holger Dammertz, Daniel Sewtz, Johannes Hanika, and Hendrik P. A. Lensch. 2010. Edge-avoiding À-Trous wavelet transform for fast global illumination filtering. In *Proc. High-Performance Graphics* (Saarbrucken, Germany). Eurographics Association, 67–75.

Paul Debevec, Tim Hawkins, Chris Tchou, Haarm-Pieter Duiker, Westley Sarokin, and Mark Sagar. 2000. Acquiring the Reflectance Field of a Human Face. In *Annual Conference Series (Proc. SIGGRAPH)*. ACM Press, 145–156. doi:10/d329r5

Xi Deng, Fujun Luan, Bruce Walter, Kavita Bala, and Steve Marschner. 2022. Reconstructing Translucent Objects Using Differentiable Rendering. In *SIGGRAPH Conference Papers*. Article 38, 10 pages. doi:10.1145/3528233.3530714

Eugene d'Eon, David Luebke, and Eric Enderton. 2007. Efficient Rendering of Human Skin. In *Rendering Techniques (Proc. EG Symposium on Rendering)* (Aire-la-Ville, Switzerland, Switzerland). Eurographics Association, 147–157. doi:10/gfzrc9

Craig Donner and Henrik Wann Jensen. 2005. Light Diffusion in Multi-Layered Translucent Materials. *ACM Trans. Graph. (Proc. SIGGRAPH)* 24, 3 (July 2005), 1032–1039. doi:10/chbmgw

Jonathan Dupuy, Eric Heitz, Jean-Claude Iehl, Pierre Poulin, Fabrice Neyret, and Victor Ostromoukhov. 2013. Linear Efficient Antialiased Displacement and Reflectance Mapping. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)* 32, 6, Article 211 (Nov. 2013), 11 pages. doi:10.1145/2508363.2508422

Elmar Eisemann and Frédo Durand. 2004. Flash Photography Enhancement via Intrinsic Relighting. 23, 3 (Aug. 2004), 673. doi:10/frbxcz

Graham Fyffe, Koki Nagano, Loc Huynh, Shunsuke Saito, Jay Busch, Andrew Jones, Hao Li, and Paul Debevec. 2017. Multi-View Stereo on Consistent Face Topology. *Comp. Graph. Forum (Proc. Eurographics)* 36, 2 (may 2017), 295–309. doi:10.1111/cgf.13127

Iliyan Georgiev, Thiago Ize, Mike Farnsworth, Ramón Montoya-Vozmediano, Alan King, Brecht Van Lommel, Angel Jimenez, Oscar Anson, Shinji Ogaki, Eric Johnston, Adrien Herubel, Declan Russell, Frédéric Servant, and Marcos Fajardo. 2018. Arnold: A Brute-Force Production Path Tracer. *ACM Trans. Graph.* 37, 3 (Aug. 2018), 32:1–32:12. doi:10/gfznb2

Abhijeet Ghosh, Graham Fyffe, Borom Tunwattanapong, Jay Busch, Xueming Yu, and Paul Debevec. 2011. Multiview Face Capture Using Polarized Spherical Gradient Illumination. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)* 30, 6 (dec 2011), 1–10. doi:10.1145/2070781.2024163

Abhijeet Ghosh, Tim Hawkins, Pieter Peers, Sune Frederiksen, and Paul Debevec. 2008. Practical Modeling and Acquisition of Layered Facial Reflectance. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)* 27, 5, Article 139 (2008), 10 pages. doi:10/drc2mb

Ioannis Gkioulekas, Anat Levin, and Todd Zickler. 2016. An evaluation of computational imaging techniques for heterogeneous inverse scattering. In *European Conference on Computer Vision (ECCV)*. Springer International Publishing, 685–701. doi:10.1007/978-3-319-46487-9_42

Ioannis Gkioulekas, Shuang Zhao, Kavita Bala, Todd Zickler, and Anat Levin. 2013. Inverse Volume Rendering with Material Dictionaries. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)* 32, 6, Article 162 (Nov. 2013). doi:10.1145/2508363.2508377

Paulo Gotardo, Jérémy Riviere, Derek Bradley, Abhijeet Ghosh, and Thabo Beeler. 2018. Practical Dynamic Facial Appearance Modeling and Acquisition. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)* 37, 6 (Dec. 2018), 232:1–232:13. doi:10/ggfg32

Kaiwen Guo, Peter Lincoln, Philip Davidson, Jay Busch, Xueming Yu, Matt Whalen, Geoff Harvey, Sergio Orts-Escolano, Rohit Pandey, Jason Dourgarian, Danhang Tang, Anastasia Tkach, Adarsh Kowdle, Emily Cooper, Mingsong Dou, Sean Fanello, Graham Fyffe, Christoph Rhemann, Jonathan Taylor, Paul Debevec, and Shahram Izadi. 2019. The Relightables: Volumetric Performance Capture of Humans with Realistic Relighting. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)* 38, 6, Article 217 (Nov. 2019), 19 pages. doi:10.1145/3355089.3356571

Hyunho Ha, Inseung Hwang, Nestor Monzon, Jaemin Cho, Donggun Kim, Seung-Hwan Baek, Adolfo Muñoz, Diego Gutierrez, and Min H. Kim. 2024. Polarimetric BSSRDF Acquisition of Dynamic Faces. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)* 43, 6, Article 275 (Nov. 2024), 11 pages. doi:10.1145/3687767

Yong He, Kayvon Fatahalian, and Tim Foley. 2018. Slang: Language Mechanisms for Extensible Real-Time Shading Systems. *ACM Trans. Graph. (Proc. SIGGRAPH)* 37, 4 (July 2018), 141:1–141:13. doi:10/gd52qd

Sebastian Herholz, Yangyang Zhao, Oskar Elek, Derek Nowrouzezahrai, Hendrik P. A. Lensch, and Jaroslav Křivánek. 2019. Volume Path Guiding Based on Zero-Variance Random Walk Theory. *ACM Trans. Graph. (Proc. SIGGRAPH)* 38, 3 (June 2019),

25:1–25:19. doi:10/ggd8m7
Homan Igehy. 1999. Tracing Ray Differentials. In *Annual Conference Series (Proc. SIGGRAPH)*. 179–186. doi:10.1145/311535.311555
Wenzel Jakob, Sébastien Speierer, Nicolas Roussel, Merlin Nimier-David, Delio Vicini, Tizian Zeltner, Baptiste Nicolet, Miguel Crespo, Vincent Leroy, and Ziyi Zhang. 2022b. *Mitsuba 3 renderer*. https://mitsuba-renderer.org.
Wenzel Jakob, Sébastien Speierer, Nicolas Roussel, and Delio Vicini. 2022a. Dr.Jit: A Just-In-Time Compiler for Differentiable Rendering. *ACM Trans. Graph. (Proc. SIGGRAPH)* 41, 4 (July 2022). doi:10.1145/3528223.3530099
Henrik Wann Jensen, Stephen R. Marschner, Marc Levoy, and Pat Hanrahan. 2001. A Practical Model for Subsurface Light Transport. In *SIGGRAPH Comput. Graph.* 511–518. doi:10.1145/383259.383319
James T. Kajiya. 1986. The Rendering Equation. *Computer Graphics (Proc. SIGGRAPH)* 20, 4 (Aug. 1986), 143–150. doi:10/cvf53j
James T. Kajiya and Brian P Von Herzen. 1984. Ray Tracing Volume Densities. *Computer Graphics (Proc. SIGGRAPH)* 18, 3 (July 1984), 165–174. doi:10/cmjk9m
Pramook Khungurn, Daniel Schroeder, Shuang Zhao, Kavita Bala, and Steve Marschner. 2015. Matching Real Fabrics with Micro-Appearance Models. *ACM Trans. Graph.* 35, 1, Article 1 (Dec. 2015), 26 pages. doi:10.1145/2818648
Alan King, Christopher Kulla, Alejandro Conty, and Marcos Fajardo. 2013. BSSRDF Importance Sampling. In *SIGGRAPH Talks*. doi:10/gfz5nz
Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proc. International Conference on Learning Representations (ICLR)*. 15 pages.
Oliver Klehm, Fabrice Rousselle, Marios Papas, Derek Bradley, Christophe Hery, Bernd Bickel, Wojciech Jarosz, and Thabo Beeler. 2015. Recent Advances in Facial Appearance Capture. *Comp. Graph. Forum (Proc. Eurographics - State of the Art Reports)* 34, 2 (May 2015), 709–733. doi:10/f7mb4b
David Koerner, Jan Novak, Peter Kutz, Ralf Habel, and Wojciech Jarosz. 2016. Subdivision Next-Event Estimation for Path-Traced Subsurface Scattering. In *EGSR Experimental Ideas and Implementations*. Eurographics Association, 6 pages. doi:10/gf6rzj
Jaroslav Křivánek and Eugene d'Eon. 2014. A Zero-Variance-Based Sampling Scheme for Monte Carlo Subsurface Scattering. In *SIGGRAPH Talks*. ACM Press, Vancouver, Canada, 66:1–66:1. doi:10/gfzq7n
Alexandr Kuznetsov, Krishna Mullia, Zexiang Xu, Miloš Hašan, and Ravi Ramamoorthi. 2021. NeuMIP: Multi-Resolution Neural Materials. *ACM Trans. Graph. (Proc. SIGGRAPH)* 40, 4, Article 175 (July 2021), 13 pages.
L. Leonard, K. Höhlein, and R. Westermann. 2021. Learning Multiple-Scattering Solutions for Sphere-Tracing of Volumetric Subsurface Effects. *Comp. Graph. Forum (Proc. Eurographics)* 40, 2 (2021), 165–178. doi:10/h2gr
Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. 2018. Differentiable Monte Carlo Ray Tracing through Edge Sampling. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)* 37, 6 (2018), 222:1–222:11. doi:10.1145/3272127.3275109
Jingwang Ling, Ruihan Yu, Feng Xu, Chun Du, and Shuang Zhao. 2024. NeRF as a Non-Distant Environment Emitter in Physics-Based Inverse Rendering. In *SIGGRAPH Conference Papers*. ACM, 1–12. doi:10/gt48w5
Stephen Lombardi, Jason Saragih, Tomas Simon, and Yaser Sheikh. 2018. Deep appearance models for face rendering. *ACM Trans. Graph. (Proc. SIGGRAPH)* 37, 4 (2018).
Guillaume Loubet, Nicolas Holzschuch, and Wenzel Jakob. 2019. Reparameterizing discontinuous integrands for differentiable rendering. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)* 38, 6 (Dec. 2019). doi:10.1145/3355089.3356510
Guillaume Loubet and Fabrice Neyret. 2017. Hybrid Mesh-Volume LoDs for All-Scale Pre-Filtering of Complex 3D Assets. *Comp. Graph. Forum* 36, 2 (2017), 431–442. doi:10.1111/cgf.13138
Wan-Chun Ma, Tim Hawkins, Pieter Peers, Charles-Felix Chabert, Malte Weiss, and Paul Debevec. 2007. Rapid Acquisition of Specular and Diffuse Normal Maps from Polarized Spherical Gradient Illumination. In *Rendering Techniques (Proc. EG Symposium on Rendering)*. doi:10.2312/EGWR/EGSR07/183-194
Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *European Conference on Computer Vision (ECCV)*. 25 pages. doi:10.1145/3503250
Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. *ACM Trans. Graph. (Proc. SIGGRAPH)* 41, 4, Article 102 (July 2022), 15 pages. doi:10.1145/3528223.3530127
Baptiste Nicolet, Alec Jacobson, and Wenzel Jakob. 2021. Large Steps in Inverse Rendering of Geometry. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)* 40, 6 (Dec. 2021). doi:10.1145/3478513.3480501
Merlin Nimier-David, Sébastien Speierer, Benoît Ruiz, and Wenzel Jakob. 2020. Radiative Backpropagation: An Adjoint Method for Lightning-Fast Differentiable Rendering. *ACM Trans. Graph. (Proc. SIGGRAPH)* 39, 4, Article 146 (July 2020), 15 pages. doi:10.1145/3386569.3392406
Marc Olano and Dan Baker. 2010. LEAN Mapping. In *Proc. Symposium on Interactive 3D Graphics and Games (I3D)*. 8 pages. doi:10.1145/1730804.1730834
Georg Petschnigg, Richard Szeliski, Maneesh Agrawala, Michael Cohen, Hugues Hoppe, and Kentaro Toyama. 2004. Digital Photography with Flash and No-Flash Image Pairs. 23, 3 (Aug. 2004), 664–672. doi:10/ffncb7
Matt Pharr, Wenzel Jakob, and Greg Humphreys. 2018. *Physically based rendering: From theory to implementation (3rd edition)*. Morgan Kaufmann.
Matt Pharr, Wenzel Jakob, and Greg Humphreys. 2023. *Physically based rendering: From theory to implementation (4th edition)*. Morgan Kaufmann.
Matt Pharr, Bartlomiej Wronski, Marco Salvi, and Marcos Fajardo. 2024. Filtering after Shading with Stochastic Texture Filtering. *Proc. ACM on Computer Graphics and Interactive Techniques* 7, 1 (May 2024), 14:1–14:20. doi:10/gtvf3c
Pramod Rao, Mallikarjun B R, Gereon Fox, Tim Weyrich, Bernd Bickel, Hans-Peter Seidel, Hanspeter Pfister, Wojciech Matusik, Ayush Tewari, Christian Theobalt, and Mohamed Elgharib. 2022. VoRF: Volumetric Relightable Faces. (2022).
Jérémy Riviere, Paulo Gotardo, Derek Bradley, Abhijeet Ghosh, and Thabo Beeler. 2020. Single-shot high-quality facial geometry and skin appearance capture. *ACM Trans. Graph. (Proc. SIGGRAPH)* 39, 4, Article 81 (Aug. 2020), 12 pages. doi:10.1145/3386569.3392464
Kripasindhu Sarkar, Marcel C. Buehler, Gengyan Li, Daoye Wang, Delio Vicini, Jérémy Riviere, Yinda Zhang, Sergio Orts-Escolano, Paulo Gotardo, Thabo Beeler, and Abhimitra Meka. 2023. LitNeRF: Intrinsic Radiance Decomposition for High-Quality View Synthesis and Relighting of Faces. In *SIGGRAPH Asia Conference Papers*. doi:10.1145/3610548.3618210
Olga Sorkine. 2006. Differential Representations for Mesh Processing. *Comp. Graph. Forum* 25, 4 (2006), 789–807. doi:10.1111/j.1467-8659.2006.00999.x
Tiancheng Sun, Kai-En Lin, Sai Bi, Zexiang Xu, and Ravi Ramamoorthi. 2021. NeLF: Neural Light-transport Field for Portrait View Synthesis and Relighting. In *EGSR - DL-only track*.
T. Tg, D. M. Tran, H. W. Jensen, R. Ramamoorthi, and J. R. Frisvad. 2024. Neural SSS: Lightweight Object Appearance Representation. *Comp. Graph. Forum (Proc. EGSR)* 43, 4 (2024). doi:10.1111/cgf.15158
Haithem Turki, Michael Zollhöfer, Christian Richardt, and Deva Ramanan. 2023. PyNeRF: Pyramidal Neural Radiance Fields. In *Advances in Neural Information Processing Systems (NeurIPS)*.
Eric Veach and Leonidas J. Guibas. 1995. Optimally Combining Sampling Techniques for Monte Carlo Rendering. In *Annual Conference Series (Proc. SIGGRAPH)*, Vol. 29. ACM Press, 419–428. doi:10/d7b6n4
Zdravko Velinov, Marios Papas, Derek Bradley, Paulo Gotardo, Parsa Mirdehghan, Steve Marschner, Jan Novák, and Thabo Beeler. 2018. Appearance Capture and Modeling of Human Teeth. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)* 37, 6 (Dec. 2018), 207:1–207:13. doi:10.1145/3272127.3275098
Dor Verbin, Ben Mildenhall, Peter Hedman, Jonathan T. Barron, Todd Zickler, and Pratul P. Srinivasan. 2024. Eclipse: Disambiguating Illumination and Materials Using Unintended Shadows. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 77–86. doi:10.1109/CVPR52733.2024.00016
Delio Vicini, Wenzel Jakob, and Anton Kaplanyan. 2021a. A Non-Exponential Transmittance Model for Volumetric Scene Representations. *ACM Trans. Graph. (Proc. SIGGRAPH)* 40, 4 (Aug. 2021), 136:1–136:16. doi:10.1145/3450626.3459815
Delio Vicini, Vladlen Koltun, and Wenzel Jakob. 2019. A Learned Shape-Adaptive Subsurface Scattering Model. *ACM Trans. Graph. (Proc. SIGGRAPH)* 38, 4 (July 2019), 15. doi:10.1145/3306346.3322974
Delio Vicini, Sébastien Speierer, and Wenzel Jakob. 2021b. Path Replay Backpropagation: Differentiating Light Paths using Constant Memory and Linear Time. *ACM Trans. Graph. (Proc. SIGGRAPH)* 40, 4 (Aug. 2021). doi:10.1145/3450626.3459804
Philippe Weier, Tobias Zirr, Anton Kaplanyan, Ling-Qi Yan, and Philipp Slusallek. 2023. Neural Prefiltering for Correlation-Aware Levels of Detail. *ACM Trans. Graph. (Proc. SIGGRAPH)* 42, 4, Article 78 (July 2023), 16 pages. doi:10.1145/3592443
Tim Weyrich, Wojciech Matusik, Hanspeter Pfister, Bernd Bickel, Craig Donner, Chien Tu, Janet McAndless, Jinho Lee, Addy Ngan, Henrik Wann Jensen, and Markus Gross. 2006. Analysis of Human Faces Using a Measurement-Based Skin Reflectance Model. *ACM Trans. Graph. (Proc. SIGGRAPH)* 25, 3 (July 2006), 1013–1024. doi:10/ctncb6
Lance Williams. 1983. Pyramidal parametrics. *SIGGRAPH Comput. Graph.* 17, 3 (July 1983), 1–11. doi:10.1145/964967.801126
Magnus Wrenninge, Ryusuke Villemin, and Christophe Hery. 2017. *Path Traced Subsurface Scattering Using Anisotropic Phase Functions and Non-Exponential Free Flights*. Technical Memo 17-07. Pixar.
Yingyan Xu, Gaspard Zoss, Prashanth Chandran, Markus Gross, Derek Bradley, and Paulo Gotardo. 2023. ReNeRF: Relightable Neural Radiance Fields with Nearfield Lighting. In *IEEE International Conference on Computer Vision (ICCV)*. 22581–22591.
Bowen Xue, Shuang Zhao, Henrik Wann Jensen, and Zahra Montazeri. 2024. A Hierarchical Architecture for Neural Materials. *Comp. Graph. Forum* 43, 6 (2024), e15116. doi:10.1111/cgf.15116
Cheng Zhang, Bailey Miller, Kai Yan, Ioannis Gkioulekas, and Shuang Zhao. 2020. Path-Space Differentiable Rendering. *ACM Trans. Graph. (Proc. SIGGRAPH)* 39, 4, Article 143 (July 2020), 19 pages. doi:10.1145/3386569.3392383
Xiuming Zhang, Sean Fanello, Yun-Ta Tsai, Tiancheng Sun, Tianfan Xue, Rohit Pandey, Sergio Orts-Escolano, Philip Davidson, Christoph Rhemann, Paul Debevec,

Jonathan T. Barron, Ravi Ramamoorthi, and William T. Freeman. 2021. Neural Light Transport for Relighting and View Synthesis. *ACM Trans. Graph.* 40, 1, Article 9 (Jan. 2021), 17 pages. doi:10.1145/3446328

Shuang Zhao, Lifan Wu, Frédo Durand, and Ravi Ramamoorthi. 2016. Downsampling Scattering Parameters for Rendering Anisotropic Media. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)* 35, 6 (2016). doi:10.1145/2980179.2980228

Yang Zhou, Tao Huang, Ravi Ramamoorthi, Pradeep Sen, and Ling-Qi Yan. 2025. Appearance-Preserving Scene Aggregation for Level-of-Detail Rendering. *ACM Trans. Graph.* 44, 1 (2025). doi:10.1145/3708343