# Inverse Transport Networks

Chengqian Che
Carnegie Mellon University

Fujun Luan
Cornell University

Shuang Zhao
University of California, Irvine

Kavita Bala
Cornell University

Ioannis Gkioulekas
Carnegie Mellon University



## Abstract

*We introduce inverse transport networks as a learning architecture for inverse rendering problems where, given input image measurements, we seek to infer physical scene parameters such as shape, material, and illumination. During training, these networks are evaluated not only in terms of how close they can predict groundtruth parameters, but also in terms of whether the parameters they produce can be used, together with physically-accurate graphics renderers, to reproduce the input image measurements. To enable training of inverse transport networks using stochastic gradient descent, we additionally create a general-purpose, physically-accurate differentiable renderer, which can be used to estimate derivatives of images with respect to arbitrary physical scene parameters. Our experiments demonstrate that inverse transport networks can be trained efficiently using differentiable rendering, and that they generalize to scenes with completely unseen geometry and illumination better than networks trained without appearance-matching regularization.*

## 1. Introduction

Acquiring models of the world has been a long-standing challenge in computer graphics and vision. During the past four decades, a variety of physics-based measurement systems [37, 58, 51, 48, 47, 3, 67] have been devised. These systems aim to reconstruct the shape and material of real-world objects under controlled or uncontrolled illumination and from images captured under a variety of radiometric devices (e.g., projector-camera systems, time-of flight devices). Following the image acquisition stage, these systems utilize an inference algorithm to solve for the physical parameters of interest, for example, shape or material. We can broadly classify such inferential algorithms into physics-based and learning-based techniques.

Physics-based techniques infer unknowns by optimizing for parameter values until they can be used, together with a physics-based image formation model, to reproduce input images. In certain cases, this optimization can be performed analytically (e.g., photometric stereo [19], direct reflectometry [37]). More generally, this optimization boils down to seeking for parameter values with which forward simulations best resemble the measurements. This framework is usually termed *analysis-by-synthesis* in computer vision or *inverse rendering* in graphics [36, 14, 71, 24, 12, 30]. By accurately modeling the underlying physics of light transport, these techniques can produce high-fidelity estimates for fully general inputs, at the cost of high, and often prohibitive, computational requirements.

Learning-based techniques, on the other hand, use supervised and unsupervised data to create functions that approximately map sensor measurements directly to physical parameters. In the last few years, predominantly these functions take the form of multi-layer neural networks, which can be efficiently optimized on large training datasets using stochastic optimization [32, 52, 57, 72, 43, 64, 31]. These techniques allow for efficient inference, but do not offer guarantees about the physical plausibility and interpretability of predicted parameter values. Additionally, since these methods are purely data-driven and do not attempt to reproduce the underlying physics, they often generalize poorly to inputs that are underrepresented in the training dataset.

We seek to combine the complementary advantages of the physics- and learning-based approaches for the inference of physical scene properties, by proposing a new physics-aware learning technique that we term *inverse transport networks*. Taking inspiration from recent work on combining physics and learning [40, 54, 33, 59, 23], inverse transport networks are trained by solving a regularized optimization problem that forces the network to produce output parameters that not only match ground-truth values but also reproduce the input images (when used as input to a forward physics-based renderer). Previously, this kind of regularized training could only be used in cases where the forward physics were sufficiently simple (i.e., can be differentiated analytically), severely restricting its applicability. By contrast, our inverse transport networks can be used with

arbitrarily complex forward physics that can capture global illumination effects such as interreflections and subsurface scattering. To this end, we introduce a new forward simulation engine that can be used for *efficient*, *general-purpose* and *physically accurate* Monte Carlo differentiable rendering. Finally, to demonstrate the general-purpose nature and improved predictive performance of inverse transport networks, we evaluate them on the task of homogeneous inverse scattering, an inverse problem involving highly multi-path and multi-bounce light transport.

## 2. Related work

**Analysis-by-synthesis in physics-based vision.** Physics-based algorithms for recovering scene parameters conceptually comprise three steps: (i) formulate an approximate image formation (or forward rendering) model as a function of the scene parameters; (ii) analytically derive an expression for the derivative of the forward model with respect to those parameters; (iii) use gradient-based optimization to solve an analysis-by-synthesis objective comparing measured and synthesized images. This approach has been used to recover shape [11, 55, 8], material [53, 41, 44], and illumination [39], either independently of each other or jointly [2, 69, 34, 35, 46].

**Differentiable rendering.** Limiting the applicability of this general approach is the need to formulate a new forward model, as well as the need to analytically compute its derivatives, specifically for each reconstruction problem. Differentiable renderers such as OpenDR [36] have been proposed as a means to remove this obstacle, by providing a general-purpose framework that can be differentiated with respect to shape, reflectance, and illumination parameters. To ensure analytical differentiability, all of the above approaches use *approximate* forward models, most often by ignoring complex light transport effects such as interreflections and subsurface scattering. This makes these methods inapplicable to situations where these effects are dominant. Some solutions have been developed for the problem of inverse scattering [14, 71, 24, 12, 30], which use physically accurate, but limited to very specialized light transport simulations, Monte Carlo differentiable renderers.

**Combining deep learning with rendering.** Recently, a number of works have emerged that propose using renderers not for analysis-by-synthesis, but as parts of learning architectures. The most popular approach is to replace the decoder network in an auto-encoder pipeline [62, 26] with a rendering layer that takes as input the parameters predicted by the encoder and produces as output synthesized images. This encoder-renderer architecture was first proposed by Wu et al. [68], who used a non-photorealistic renderer to achieve categorical interpretability (e.g., "a boy and girl stand next to a bench"). Similar architectures have subsequently been proposed for physics-based inference of parameters such as surface normals, illumination, and reflectance [40, 54, 33, 59, 23]. Alternatively, renderers have been incorporated into adversarial learning pipelines for image-to-image translation tasks [60].

## 3. Our Approach

**Problem setting.** We are interested in inverse problems where the unknowns are physical properties (e.g., geometry, material (optical) parameters, and illumination) of a scene imaged by a radiometric sensor. Each image captured by the sensor records photons interacting multiple times with the surfaces and interior of objects in the scene, in a way that depends on the scene parameters. We will refer to this complex *light transport* process using $\mathcal{T}(\boldsymbol{\pi})$, where $\boldsymbol{\pi}$ are the relevant physical parameters of the scene: $\boldsymbol{\pi} = \{\text{sensor}, \text{geometry}, \text{material}, \text{illumination}\}$.

A long-standing approach for solving such inverse problems in computer vision and graphics is *analysis by synthesis*, also known as *inverse rendering*. Given image measurements $I$, we search for parameters $\boldsymbol{\pi}$ that, when used to synthesize images, can closely match the measurements:

$$\hat{\boldsymbol{\pi}} = \underset{\boldsymbol{\pi}}{\operatorname{argmin}} \left\| I - \mathcal{T}(\boldsymbol{\pi}) \right\|^2. \tag{1}$$

Analysis by synthesis offers two key advantages that make it an attractive algorithm for solving inverse problems of this kind. First, it is a general-purpose procedure that can be applied to arbitrary scenes. This is thanks to the advent in computer graphics of *forward rendering* algorithms that can accurately simulate light transport $\mathcal{T}(\boldsymbol{\pi})$ of arbitrary complexity, including global illumination effects such as interreflections, specular and refractive caustics, and multiple scattering. Second, it is often possible to derive guarantees about the fidelity of the reconstructed parameters $\hat{\boldsymbol{\pi}}$, through analysis of the underlying physics. Unfortunately, solving optimization problem (1) is often a computationally intensive process: even when it is possible to compute derivatives $\partial\mathcal{T}(\boldsymbol{\pi})/\partial\boldsymbol{\pi}$ for gradient descent optimization, finding a (local) minimum of Equation (1) requires performing thousands of expensive rendering operations.

An alternative methodology for solving such inverse problems is through supervised learning. Given a training set of image measurements $\{I_d\}$ and groundtruth parameters $\{\boldsymbol{\pi}_d\}$, $d = 1, \ldots, D$, we first use empirical risk minimization to train a parametric regression model $\mathcal{N}[\boldsymbol{w}]$, e.g., a neural network, that directly maps images to parameters:

$$\hat{\boldsymbol{w}} = \underset{\boldsymbol{w}}{\operatorname{argmin}} \sum_{d=1}^{D} \left\| \boldsymbol{\pi}_d - \mathcal{N}[\boldsymbol{w}](I_d) \right\|^2. \tag{2}$$

Once the trained network $\mathcal{N}[\hat{\boldsymbol{w}}]$ is available, we can use it to efficiently obtain estimates $\hat{\boldsymbol{\pi}}$ for the physical parameters underlying new image measurements $I$, by performing a *forward pass* operation: $\hat{\boldsymbol{\pi}} = \mathcal{N}[\hat{\boldsymbol{w}}](I)$. This efficiency comes with the caveat that it is difficult to obtain guarantees about the quality of the estimates $\hat{\boldsymbol{\pi}}$. This becomes particularly important when we use the network to process images of scenes that are not well represented in the training set, e.g., very different shapes or illumination. Given the highly nonlinear mapping $\mathcal{T}$ from scene to images, it can be challenging for networks to generalize to unseen scenes.

### 3.1. Inverse Transport Networks

We seek to combine the efficiency of learning with the generality of analysis by synthesis. For this, we propose to regularize the loss function (2) used to train a regressor network with a term that closely resembles the loss function (1) optimized by analysis by synthesis:

$$\hat{\boldsymbol{w}} = \underset{\boldsymbol{w}}{\operatorname{argmin}} \sum_{d=1}^{D} \Bigg[ \underbrace{\left\| \boldsymbol{\pi}_d - \mathcal{N}[\boldsymbol{w}](I_d) \right\|^2}_{\text{supervised loss}} + \lambda \underbrace{\left\| I_d - \mathcal{T}\left(\mathcal{N}[\boldsymbol{w}](I_d)\right) \right\|^2}_{\text{regularization}} \Bigg]. \tag{3}$$

The regularization term in Equation (3) forces the neural network to predict parameters $\boldsymbol{\pi}_d$ that not only match the groundtruth, but also can reproduce the input images when used as input to forward rendering. This has two desirable effects as follows. First, the parameters predicted by the network are likely to be close to those that would be obtained from analysis-by-synthesis or inverse rendering, as the regularization term in Equation (3) is equivalent to the analysis-by-synthesis loss (1). Second, the regularization term forces the regression function $\mathcal{N}[\hat{\boldsymbol{w}}]$ implemented by the neural network to be approximately equal to the inverse of the light transport operator $\mathcal{T}$, that is, $\mathcal{N}[\hat{\boldsymbol{w}}] \approx \mathcal{T}^{-1}$. Given that $\mathcal{T}$ models the physics of light transport that apply generally to all possible scenes, we expect the resulting neural network generalize well to novel scenes. Accordingly, we term networks trained with the loss function (3) as *inverse transport networks* (ITN). In practice, our ITNs can be implemented using the encoder-decoder architecture with the decoder replaced with physics-based renderers. A visualization of this architecture is shown in Figure 1(b).

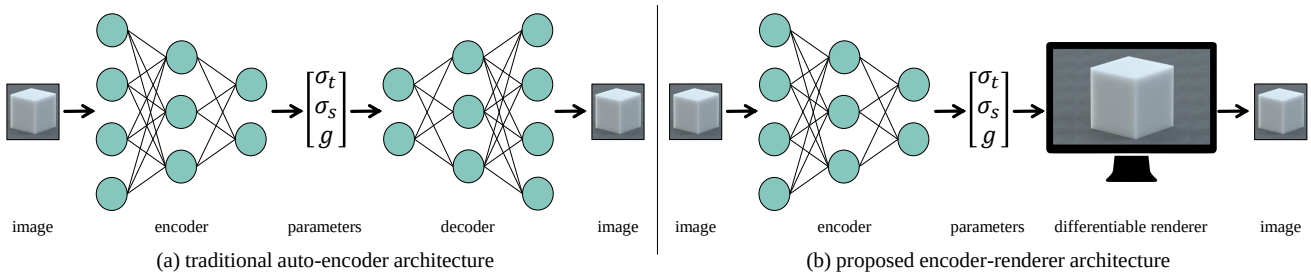
**Training inverse transport networks.** Using our ITNs requires overcoming a key computational challenge, namely solving the optimization problem (3) for their training. The difficulty comes exactly from the light transport operator $\mathcal{T}$, whose evaluation generally requires solving the radiative transfer [6] and rendering equations [21]. As computer graphics provides us with rendering algorithms for approximate forward evaluation of $\mathcal{T}$, training could be performed using algorithms such as REINFORCE [68], without differentiating the regularization term. However, such algorithms are known to suffer from slow convergence.

Instead, it would be preferable to optimize the loss (3) using state-of-the-art stochastic gradient descent algorithms [56, 25, 70, 9], also known as backpropagation [4, 29, 27]. This requires being able to estimate derivatives of the light transport operator $\mathcal{T}$ with respect to physical parameters $\boldsymbol{\pi}$ in an unbiased manner, a task we refer to as *differentiable rendering*. Existing differentiable rendering engines such as OpenDR [36] can do this computation only for approximate direct lighting models. This goes against our goal for training networks by accounting for the full light transport process. Instead, in the next section, we address this computational problem by developing an *efficient*, *general-purpose* and *physically accurate* differentiable rendering engine, based on Monte Carlo integration.

**Post-learning refinement.** At test time, given input images, performing a formward pass through the trained network $\mathcal{N}$ provides us with a first estimate of the unknown parameters $\boldsymbol{\pi}$ underlying an input image $I$. We can further improve the parameter estimate through a second estimation stage, where we use stochastic gradient descent together with Monte Carlo differentiable rendering to minimize the analysis-by-synthesis loss (1) for the image $I$. Critically, we can initialize this second estimation stage using the parameter estimate produced by the network.

The effect of this seeding is that the analysis-by-synthesis optimization can converge to a solution much faster than if we had skipped the network-based estimation stage and used a random initial point. We expect the ITN architecture to be particularly effective for this kind of analysis-by-synthesis acceleration, given that the regularization term in its training loss function (3) encourages the network to produce estimates that are close to those that would be obtained by directly performing analysis-by-synthesis optimization. Additionally, as the network is trained using supervised information, we expect that this initialization will help the optimization procedure converge to the global minimum of the analysis-by-synthesis loss (1), avoiding ambiguities in the physical parameter space.

**Relationship to prior work.** Regularization similar to Equation (3) have previously appeared in two types of literature. The first is autoencoder architectures [62, 26] that, in addition to the regressor (*encoder*) network $\mathcal{N}[\boldsymbol{w}]$ (which maps images to parameters), uses a second *decoder* network $\mathcal{D}[\boldsymbol{u}]$ that maps the parameters back to images. Then, the regularization term in Equation (3) is replaced with $\|I_d - \mathcal{D}[\boldsymbol{u}](\mathcal{N}[\boldsymbol{w}](I_d))\|^2$, and both the encoder and decoder networks are trained simultaneously. These architectures are of great utility when seeking to infer semantic parameters (e.g., a class label) about a scene, in which




(a) traditional auto-encoder architecture (b) proposed encoder-renderer architecture

Figure 1. **Inverse transport networks:** (a) Traditional autoencoder network architectures use two neural networks, encoder and decoder, to learn intermediate representations of input images that are subsequently remapped into images. (b) Our proposed inverse transport network architecture replaces the decoder with a differentiable physics-based renderer, which improves generalization performance by acting as a regularization term on the encoder.

case there is often no analytical model for the forward mapping of these parameters to images. However, when the unknowns $\boldsymbol{\pi}$ are physical parameters, autoencoder architextures do not take advantage of the rich knowledge we have from the physics governing the forward operator $\mathcal{T}$. Additionally, the learned forward mapping $\mathcal{D}[\boldsymbol{u}]$ may not generalize to unseen instances, as it is specific to the data set used to train the autoencoder. Figure 1 compares the autoencoder and inverse transport architectures.

The second type of architectures are networks using regularization terms with the light transport operator $\mathcal{T}$ approximated by some $\mathcal{S}$ [40, 54, 33, 59, 23]. These approximations are generally based on direct lighting models for image formation, where photons are assumed to only interact with the scene once between leaving a light source and arriving at a detector (e.g., direct reflection without interreflections, single scattering). Consequently, these networks perform suboptimally in cases where higher-order transport effects are predominant. Inspired by these prior works, our ITNs are general physics-aware learning pipelines that can be used for solving inverse problems involving light transport of arbitrary complexity. To demonstrate the importance of using the full light transport operator $\mathcal{T}$ instead of direct-lighting approximations $\mathcal{S}$, we compare in Section 5 these approaches as well as the unregularized approach of Equation (2), for the problem of inferring scattering parameters of translucent materials. We choose this application as an instance of an extreme multi-path, multi-bounce transport problem, where single scattering approximations have limited applicability.

## 4. Differentiable Monte Carlo Rendering

**Background.** We aim to develop algorithms for estimating derivatives of radiometric quantities, e.g., intensity measured by a detector, with respect to physical scene properties, e.g., optical material of objects in the scene, in an unbiased manner. Our formulation directly borrows from the path integral formulation for forward rendering. Therefore to make our discussion self-contained, we provide here the necessary background. Our starting point is the expression of radiometric quantities as integrals over the space of possible light particle paths [61]:

$$\mathcal{T}(\boldsymbol{\pi}) = \int_{\mathbb{P}} f[\boldsymbol{\pi}](\bar{\boldsymbol{x}})\, \mathrm{d}\bar{\boldsymbol{x}}, \tag{4}$$

where, for any $K > 1$, $\bar{\boldsymbol{x}} := (\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_K)$ with $\boldsymbol{x}_i \in \mathbb{R}^3$ (for $i = 0, 1, \ldots, K$) indicates a light transport path with $\boldsymbol{x}_0$ located on a light source and $\boldsymbol{x}_K$ on a sensor, and

$$f(\bar{\boldsymbol{x}}) = G[\boldsymbol{\pi}](\boldsymbol{x}_{K-1}, \boldsymbol{x}_K) \prod_{k=1}^{K-1} G[\boldsymbol{\pi}](\boldsymbol{x}_{k-1}, \boldsymbol{x}_k)\, f_s[\boldsymbol{\pi}](\boldsymbol{x}_{k-1}, \boldsymbol{x}_k, \boldsymbol{x}_{k+1}). \tag{5}$$

This integration is performed over the space $\mathbb{P}$ of all possible paths. In each such path, the intermediate points $\boldsymbol{x}_k$ with $0 < k < K$ capture light-scene interactions via reflection, refraction, and subsurface scattering. The *throughput function* $f[\boldsymbol{\pi}]$ describes the amount of radiance contributed by a path as a function of the scene geometry, material properties, illumination and detector characteristics. The throughput can be expressed as the product of per-vertex terms $G[\boldsymbol{\pi}]\, f_s[\boldsymbol{\pi}]$, whose exact role varies for different points $\boldsymbol{x}_k$. In particular, when $\boldsymbol{x}_k$ is a point on the surface of an object, then $f_s[\boldsymbol{\pi}]$ is equal to the bidirectional scattering distribution function (BSDF) of the object at point $\boldsymbol{x}_k$ with normal $n(\boldsymbol{x}_k)$. If one of $\boldsymbol{x}_{k-1}$ and $\boldsymbol{x}_{k+1}$ is outside the object and the other inside, then $f_s[\boldsymbol{\pi}]$ describes a refraction event; otherwise it describes a reflection event. If $\boldsymbol{x}_k$ is a point inside a scattering medium, then $f_s[\boldsymbol{\pi}]$ describes a scattering event and is equal to the product of the medium's volumetric albedo and phase function at $\boldsymbol{x}_k$. In both cases, the BSDF and phase function are evaluated on incoming and outgoing directions $\boldsymbol{x}_{k-1} \to \boldsymbol{x}_k$ and $\boldsymbol{x}_k \to \boldsymbol{x}_{k+1}$, respectively. Finally, the term $G[\boldsymbol{\pi}]$ equals binary visibility when both $\boldsymbol{x}_{i-1}$ and $\boldsymbol{x}_i$ are outside scattering media. Otherwise, it equals volumetric attenuation, and is

a function of the medium's extinction coefficient. A visualization of this is shown in Figure 2(a).

The path integral formulation of Equation (4) accurately describes light transport for scenes of arbitrary complexity, including higher-order transport effects such as interreflections and multiple scattering that cannot be represented using direct lighting approximations. Unfortunately, except for trivial scenes, Equation (4) cannot be evaluated analytically. Computer graphics has focused on the approximation of Equation (4) using Monte Carlo integration [10, 61, 49], which at a high-level operates as follows. First, a set of paths $\{\bar{\boldsymbol{x}}_n : n = 1, \ldots, N\}$ are sampled stochastically from a probability density $p$ defined on the path space $\mathbb{P}$. Second, the throughput $f_s[\boldsymbol{\pi}]$ of each these paths is computed. Third and final, an unbiased and consistent estimator of Equation (4) is formed as:

$$\langle \mathcal{T}(\boldsymbol{\pi}) \rangle = \frac{1}{N} \sum_{n=1}^{N} \frac{f[\boldsymbol{\pi}](\bar{\boldsymbol{x}}_n)}{p(\bar{\boldsymbol{x}}_n)}. \tag{6}$$

The performance of these estimators depends critically on the propability distribution $p$ used to sample light transport paths. Modern Monte Carlo rendering techniques use path sampling techniques such as path tracing [22], bidirectional path tracing (BDPT) [28], and Metropolis light transport (MLT) [61], which simulate path sampling densities $p$ similar to the throughput function $f_s[\boldsymbol{\pi}]$ in order to reduce the variance of the estimator (6).

### 4.1. Differentiable rendering

Instead of image measurements $\mathcal{T}(\boldsymbol{\pi})$, we focus on estimating their derivatives $\partial\mathcal{T}(\boldsymbol{\pi})/\partial\boldsymbol{\pi}$ with respect to scene parameters $\boldsymbol{\pi}$ describing illumination, geometry, optical material. To this end, we follow previous work [14, 71, 24, 12] that generalizes the path integral formulation to apply for such derivatives. In this setting, differentiating Equation (4) and rearranging the throughput terms yields:

$$\partial\mathcal{T}(\boldsymbol{\pi})/\partial\boldsymbol{\pi} = \int_{\mathbb{P}} f[\boldsymbol{\pi}](\bar{\boldsymbol{x}})\, S[\boldsymbol{\pi}](\bar{\boldsymbol{x}})\, \mathrm{d}\bar{\boldsymbol{x}}, \tag{7}$$

$$\text{where} \quad S[\boldsymbol{\pi}](\bar{\boldsymbol{x}}) = \sum_{k=1}^{K-1} S_s[\boldsymbol{\pi}](\boldsymbol{x}_{k-1}, \boldsymbol{x}_k, \boldsymbol{x}_{k+1}), \tag{8}$$

$$S_s[\boldsymbol{\pi}](\boldsymbol{x}_{k-1}, \boldsymbol{x}_k, \boldsymbol{x}_{k+1}) = \frac{\partial f_s[\boldsymbol{\pi}](\boldsymbol{x}_{k-1}, \boldsymbol{x}_k, \boldsymbol{x}_{k+1})/\partial\boldsymbol{\pi}}{f_s[\boldsymbol{\pi}](\boldsymbol{x}_{k-1}, \boldsymbol{x}_k, \boldsymbol{x}_{k+1})}. \tag{9}$$

Compared to Equation (4), the path integral for the derivative case includes the *score function* $S[\boldsymbol{\pi}]$, that sums derivatives of the per-vertex throughput with respect to scene parameters $\boldsymbol{\pi}$. Similar to the forward rendering case, we can combine this path integral expression with Monte Carlo integration, in order to form a consistent and unbiased estimate of the derivative as:

$$\langle \partial\mathcal{T}(\boldsymbol{\pi})/\partial\boldsymbol{\pi} \rangle = \frac{1}{N} \sum_{n=1}^{N} \frac{f[\boldsymbol{\pi}](\bar{\boldsymbol{x}}_n)\, S[\boldsymbol{\pi}](\bar{\boldsymbol{x}}_n)}{p(\bar{\boldsymbol{x}}_n)}. \tag{10}$$

When implementing Monte Carlo differentiable rendering, an important challenge is computing the derivative terms involved in the evaluation of the score function $S[\boldsymbol{\pi}]$. In several cases, it is possible to derive analytical expressions for the derivatives of the throughput function $f[\boldsymbol{\pi}]$ with respect to certain scene parameters $\boldsymbol{\pi}$. This approach has been used previously to estimate derivatives with respect to scattering material parameters [14, 71, 24, 12].

Here we take a different approach: Instead of hard-coding derivatives for a pre-defined set of parameters, we combine a general-purpose Monte Carlo renderer with automatic differentiation [15, 50]. This allows us to compute physically accurate derivatives of the throughput function $f[\boldsymbol{\pi}]$ with respect to arbitrary scene parameters.

**Implementation.** We have developed a new simulation engine named *MitsubaDR* that integrates the Stan Math Library [5] for automatic differentiation, with the Mitsuba engine [20] for physically accurate Monte Carlo rendering. MitsubaDR supports the same variety of materials, light sources, and camera models as the original Mitsuba renderer, and currently supports differentiation with respect to the following types of scene parameters:

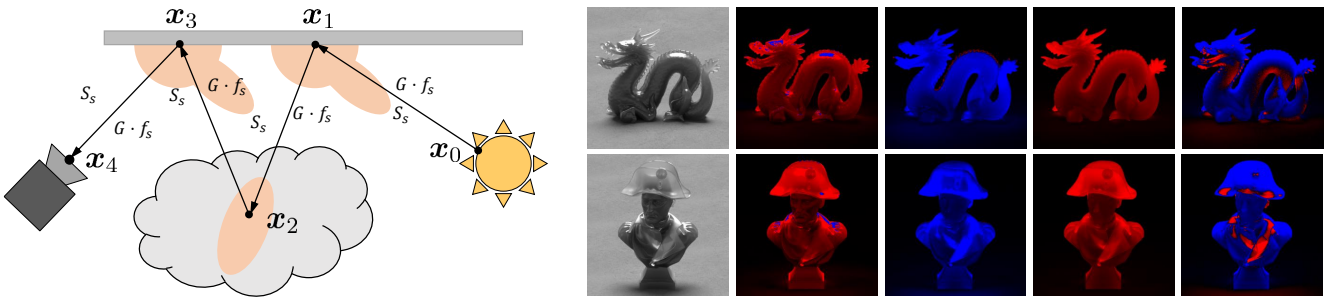
- Spatially constant and varying BSDF models, including ideal Lambertian and specular reflectance, physics-inspired microfacet models [7] for rough diffuse [45] and specular [63, 66, 1] reflectance and refraction, as well as dictionary BSDF representations [38]. For each of these models, differentiation is possible with respect to their albedo, roughness, index of refraction, or mixing weight parameters, as applicable.

- Spatially constant and varying scattering, including a variety of phase function models such as Henyey-Greenstein [16], von-Mises Fisher [13], and their linear combinations. Differentiation is possible with respect to the extinction coefficient, single-scattering albedo, and phase function parameters.

- Geometry, such as spatially varying bump maps and normal maps that can be differentiated with respect to depth and normal displacements, respectively.

- Environment map illumination, including the Hoek-Wilkie sky and sun-sky models [18, 17], and dictionary representations such as spherical harmonics.

Figure 2b shows examples of rendered image derivatives with respect to a few of these parameters. Additionally,




(a) forward and differentiable Monte Carlo rendering

(b) example forward and derivative renderings

Figure 2. **Monte Carlo forward and differentiable rendering:** (a) Monte Carlo rendering estimates radiometric measurements by randomly sampling photon paths $\bar{\boldsymbol{x}}$ and aggregating their radiometric throughput $f$. By additionally evaluating and aggregating the radiometric score $S$ for each path, we can use the same procedure to additionally estimate *derivatives* of radiometric measurements with respect to physical scene parameters. (b) Example forward and differentiable renderings for two different scenes. In each scene, the object's surface reflectance and transmittance are characterized by a rough dielectric BSDF with roughness parameter $r$, and its subsurface scattering is characterized by spatially-homogeneous extinction coefficient $\sigma_t$, volumetric albedo $\alpha$, and Henyey-Greenstein phase function with some value $g$. For each of the two objects, we show from left to right: a forward rendering, and rendered derivatives with respect to $r$, $\sigma_t$, $\alpha$, and $g$. In the differentiable renderings, red indicates positive and blue negative values. Additionally, most of the background in the differentiable renderings is black, as the intensities at those parts of the image are largely independent of the object's material parameters.

MitsubaDR is designed to be easily extensible for differentiation with respect to other scene properties. Additional capabilities, such as new BSDF models, can be incorporated using the plugin system of Mitsuba.

**Path sampling algorithms.** Although the differentiated estimator (10) has a similar form to the original one (6), the difference between $f[\boldsymbol{\pi}]$ and $f[\boldsymbol{\pi}]S[\boldsymbol{\pi}]$ suggests that efficient derivative estimation would require the development of new path sampling techniques that generate paths from probability distributions $p$ that approximate $f[\boldsymbol{\pi}]S[\boldsymbol{\pi}]$. However, it has been observed empirically that derivative estimation can still be done efficiently using the same path sampling techniques as for forward rendering [71, 24, 12]. MitsubaDR follows this approach, and uses path tracing for both forward and differentiable rendering.

**Stochastic optimization.** In addition to its generality and physical accuracy, Monte Carlo differentiable rendering provides computational advantages in the context of gradient-based optimization. In particular, learning deep neural networks from large training datasets using empirical risk minimization, as in Equation (2), greatly relies on the ability to perform backpropagation in a *stochastic* manner. Namely, one can compute derivatives of the loss function using stochastic subsets of the training set (*minibatches*). Changing the size of the minibatches allows controlling the tradeoff between the cost of gradient computations and the number of iterations until convergence [4, 29, 27].

Monte Carlo differentiable rendering offers control over a similar capability: we can reduce the number of sampled paths to speedup derivative computation at the cost of increased variance. As the derivative estimations (10) remain consistent and unbiased, we can use this to take advantage of the same convergence guarantees and tradeoffs as with stochastic backpropagation. Therefore, our Monte Carlo differentiable rendering engine is particularly well-suited for training of neural networks using state-of-the-art stochastic gradient descent algorithms [56, 25, 70, 9].

## 5. Application: Inverse Scattering

To demonstrate the applicability and utility of our proposed ITN architecture, we focus on the problem of *homogeneous inverse scattering*: given an image of a translucent object of known shape under known illumination, we aim to determine the optical material parameters that control the scattering of light inside this object. Specifically, the material is characterized by a triplet of macroscopic bulk parameters $\boldsymbol{\pi} = \{\sigma_t, \alpha, f_p\}$ as follows.

- The *extinction coefficient* $\sigma_t$ is a scalar corresponding to the optical density of the material and controls the average distance between consecutive volume events (also known as *mean free path*).
- The *volumetric albedo* $\alpha$ is a scalar probability and controls whether a photon is scattered or absorbed at a volume event.
- The *phase function* $f_p$ is a probability distribution over the sphere of directions and controls the direction scattered photons continue to travel towards.

In general, these parameters can be spatially varying, but in our application we assume them to remain constant everywhere inside the object (homogeneous scattering).

Inverse scattering is a problem that is particularly well-suited for ITNs because of the characteristic complexity of light transport within translucent objects. Each photon propagating inside a scattering medium undergoes a random walk, controlled non-linearly by the medium's bulk parameters. These random walks typically involve more than one bounces. In turn, a radiometric detector capturing an image of such an object accumulates a large number of photons, each performing a different random walk. Therefore, images of translucent objects are a typical example involving extremely multi-path and multi-bounce light transport. Although it is possible to simplify this image formation process by assuming that each photon only bounces once inside the object, this *single-scattering* approximation is only applicable to very optically thin materials such as diluted liquids [42], necessitating the development of general-purpose inverse scattering techniques to accurately model the full complexity of volumetric light transport [14].

**Loss functions and architectures.** We use the homogeneous inverse scattering setting to compare neural networks trained with three different loss functions. The first network is trained using the purely supervised loss (2). We refer to this as the *regressor network* (RN). The second network is trained using the regularized loss (3), where the $\mathcal{T}$ corresponds to simulating the full volumetric light transport. This is the *inverse-transport network* (ITN) introduced in Section 3.1. The third network is trained again using the regularized loss (3), but this time with $\mathcal{T}$ replaced with the single-scattering approximation $\mathcal{S}$ to volumetric light transport. We term this the *single-scattering network* (SSN).

For all the three networks, we use the structure proposed by Liu et al. [33]. Specifically, each network is composed of seven convolutional layers, and the size of the output channel for each layer is reduced to half the size of its input. Each convolutional layer is followed by a rectified linear unit (ReLU) and a max-pooling layer. A fully connected layer follows the convolutional layers at the end.

For both the ITN and SSN, we perform forward and differential evaluations of the transport operators $\mathcal{T}$ and $\mathcal{S}$, respectively, using MitsubaDR. The rendering layer is connected to the network and takes as input the material parameters $\boldsymbol{\pi} = \{\sigma_t, \alpha, f_p\}$ predicted by the network, as shown in Figure 1(b). These parameters are used to render images and their per-parameter derivatives, as needed for evaluating the loss function and performing backpropagation.

**Datasets and evaluation.** For our quantitative comparisons, we use a synthetic dataset containing images of translucent objects with varying geometry, illumination, and scattering parameters. We use ten different object shapes, as shown in Figure 3, selected among common computer graphics meshes. Each shape is placed under ten different illumination conditions created using the Hošek-Wilkie sun-sky model [18, 17]. Finally, for each shape and illumination combination, we render images for different scattering parameter triplets $\boldsymbol{\pi}$, spanning the following ranges for each individual parameter: $\sigma_t \in \left[25\,\mathrm{mm}^{-1}, 300\,\mathrm{mm}^{-1}\right]$, $\alpha \in [0.3, 0.95]$, and Henyey-Greenstein phase functions $f_p$ with parameter $g \in [0, 0.9]$. This covers materials with translucent appearance ranging from near-transparent to near-opaque. We use the Mitsuba physics-based renderer [20] to simulate $20,000$ high-dynamic range images under these settings.

We use the images corresponding to six shapes and four illuminations as the training set, and use all of the remaining images for testing. This yields a testing set that includes images of objects of unseen shape, or under unseen illumination, or both. We use this separation between training and testing images specifically in order to evaluate the generalization properties of the three different neural network architectures. Specifically, we compare the RN, ITN and SSN in terms of how accurately they can predict material parameters, how accurately they can reproduce input image appearance, and how much they can accelerate analysis-by-synthesis, for images from both the training and test sets.

## 6. Experiments

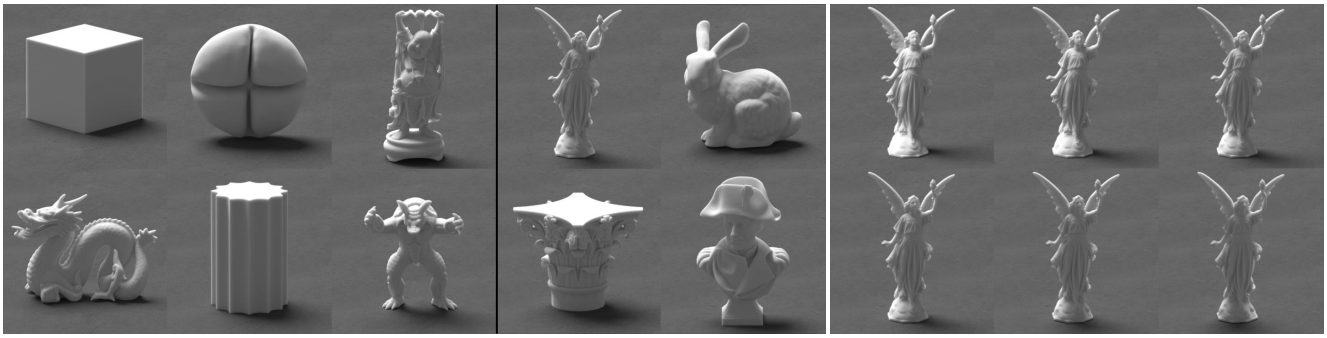
When training the ITN and SSN, we use as initialization a network trained with only the supervised loss of Equation (3) for few epochs. We set $\lambda$ in Equation (3) so that the supervised loss and regularization term for both ITN and SSN have approximately the same magnitude. All networks are trained using the Adam optimization algorithm [25].

We evaluate the three networks, RN, ITN and SSN, in three ways: First, we consider how accurately they predict the material parameters unrelying input images. Second, we examine how well images rendered with the predicted parameters match the appearance of the corresponding input images. Third, we evaluate the utility of parameter predictions for initializing inverse rendering optimization problems of the form of Equation (1).

| network | training | | | testing | | |
|---|---|---|---|---|---|---|
| | $\sigma_t$ | $\alpha$ | $g$ | $\sigma_t$ | $\alpha$ | $g$ |
| RN | **24.43** | **0.06** | **0.13** | 81.59 | 0.15 | 0.41 |
| ITN | 41.80 | 0.08 | 0.25 | **57.44** | **0.10** | **0.25** |
| SSN | 56.50 | 0.17 | 0.34 | 65.06 | 0.18 | 0.35 |

Table 1. Average RMSE of parameters predicted using RN, ITN and SSN.

**Parameter prediction.** For this evaluation, we use RN, ITN and SSN to predict three material parameters $\boldsymbol{\pi} = \{\sigma_t, \alpha, g\}$ for input images coming from both the training and testing datasets. We compare them with ground-truth values and present the average root-mean-square er-



training set testing set

(a) shapes used in dataset (b) illuminations used in dataset

Figure 3. **Dataset for homogeneous inverse scattering:** We render 20,000 images by combining different shape, illumination, and scattering material conditions. (a) For shape, we use ten meshes commonly encountered in the computer vision and graphics literature. We split these into shapes used only for training, and shapes used only for testing. (b) For illumination, we use the Hošek-Wilkie sun-sky model under different orientations, corresponding to several illumination directions between side-lighting and back-lighting.

ror (RMSE) in Table 1. We observe that, even though the RN can achieve lower prediction error on the training set, the ITN performs better on the testing set. These results provides evidence that the regularization term in the Equation (3) used for training the ITN allows the network to generalize better to new scenes with unsheen shapes and illumination. Additionally, by considering the performance of the SSN on the testing set, we observe that it is not sufficient to use the single-scattering operator $\mathcal{S}$ (as in SSN). Instead, better generalization requires using the full volumetric light transport operator $\mathcal{T}$ (as in ITN).

| net- | training | | testing | |
|---|---|---|---|---|
| work | RMSE | 1-MS-SSIM | RMSE | 1-MS-SSIM |
| RN | 0.3368 | 0.0665 | 0.3180 | 0.1002 |
| ITN | **0.3299** | **0.0609** | **0.1631** | **0.0426** |
| SSN | 0.9444 | 0.1673 | 0.8735 | 0.1585 |

Table 2. Average RMSE and $1-$ MS-SSIM of image appearance predicted using RN, ITN and SSN.

**Appearance reproduction.** To evaluate the networks in terms of their ability to reproduce input image appearance, we use the material parameters predicted by the RN, ITN and SSN to render images with the same shape and illumination settings as what is used for the input image. We then compare the rendered and input images using two different metrics: (1) RMSE, and (2) the multi-scale structural similarity index (MS-SSIM) [65], as a representative perceptually-motivated image quality metric. We average each of these metrics across both the training and testing set, as shown in Table 2. (Note that we report $1-$MS-SSIM instead of MS-SSIM values, to ensure that across all metrics lower values indicate better performance.)

We note that, unlike the case of parameter prediction (Table 1), here the ITN outperforms the RN in terms of appearance prediction on both the training and testing set. This improved performance is particularly pronounced in the case of the unseen scenes in the testing set, where the ITN results in an improvement in appearance prediction of about $50\%$. In contrast, SSN has the highest appearance errors in both training and testing sets. This suggests that the single-scattering approximation employed by the SSN is not sufficient for accurately reproducing input images whose appearance is the result of higher-order light transport.

| | | number of iterations | | | | |
|---|---|---|---|---|---|---|
| | | 1 | 50 | 100 | 150 | 200 |
| loss | RN | 0.0670 | 0.0041 | 0.0023 | 0.0018 | 0.0015 |
| | ITN | **0.0189** | **0.0024** | **0.0019** | **0.0015** | **0.0013** |
| $\log(\sigma_t)$ | RN | **0.1506** | **0.0488** | 0.0346 | 0.0265 | 0.0224 |
| | ITN | 0.2084 | 0.0805 | **0.0191** | **0.0090** | **0.0047** |
| $\alpha$ | RN | 0.0092 | **0.0009** | **0.0006** | 0.0006 | 0.0005 |
| | ITN | **0.0039** | 0.0010 | 0.0008 | **0.0003** | **0.0002** |
| $g$ | RN | 0.0812 | 0.0404 | 0.0205 | 0.0121 | 0.0069 |
| | ITN | **0.0354** | **0.0276** | **0.0143** | **0.0065** | **0.0031** |

Table 3. Medians MSE of image loss, $\log(\sigma_t)$ , $\alpha$ and $g$ for both RN and ITN

**Initialization of inverse rendering.** Finally, we compare the parameter predictions of the RN and ITN in terms of how much they accelerate analysis-by-synthesis when used to initialize optimization problems of the form of Equation (1). For this, we randomly select 10 images rendered with each shape in the test set, and use them to perform inverse rendering. We use the parameters predicted by RN and ITN to initialize the inverse rendering process, which is run for a fixed number of 200 stochastic gradient descent iterations. Every 50 iterations (including at the initialization), and we record the median value of the appearance loss used

for the optimization (Equation (1)), as well as the errors between the groundtruth parameters and the parameters at the end of that iteration.

From Table 3, we observe that the parameter predictions produced by the ITN are more useful for initialization, as they result in lower values of the loss function after the same number of iterations. Additionally, we observe that the parameters the inverse rendering procedure converges to are closer to the groundtruth when using ITN-based initialization: Before any optimization iterations, all three parameters are initialized to be closer to the ground-truth value except for $\log(\sigma_t)$. After a fixed number of iterations, the median parameter errors of the procedures initialized using ITN are uniformly smaller than those initialized using RN.

## 7. Conclusions

We introduced inverse transport networks as a new neural network architecture that can be used in inverse problems where it is necessary to predict *physical* parameters (shape, material, illumination) underlying some input images. These networks are trained so that their parameter predictions not only approximate groundtruth parameters (supervised loss), but also can be used to synthesize images that closely match the corresponding input images (unsupervised regularization). Our experiments show that, when image synthesis is performed using physically-accurate rendering algorithms that capture all light transport effects in the input images, this regularization significantly improves the generalization of trained networks to inputs consisting of previously unseen shapes, illumination, or both.

This improved generalization performance comes at a high computational cost, due to the need to minimize a training function that includes forward Monte Carlo rendering operations. We alleviated this cost by introducing a general-purpose, physically-accurate *differentiable renderer*. This renderer allows us to estimate derivatives of images with respect to physical scene parameters, which in turn means that we can use efficient stochastic gradient descent procedures to train the inverse transport networks.

By demonstrating the utility of inverse transport networks for physical inference tasks, and of differentiable renderers for training such networks, we hope to motivate future work in both computer graphics and computer vision. In computer graphics, an exciting direction of research is the development of optimal path sampling techniques for differentiable rendering, which so far has received limited attention [14]. In computer vision, the development of differentiable renderers opens up a whole new direction of exploration, as researchers investigate more general learning architectures that intelligently combine neural networks with physics-based simulation.

## References


[1] M. Ashikhmin and P. Shirley. An anisotropic phong brdf model. *Journal of graphics tools*, 5(2):25–32, 2000.

[2] J. T. Barron and J. Malik. Shape, illumination, and reflectance from shading. *IEEE transactions on pattern analysis and machine intelligence*, 37(8):1670–1687, 2015.

[3] M. Bertero, T. A. Poggio, and V. Torre. Ill-posed problems in early vision. *Proceedings of the IEEE*, 76(8):869–889, 1988.

[4] L. Bottou and O. Bousquet. The Tradeoffs of Large Scale Learning. *NIPS*, 2008.

[5] B. Carpenter, M. D. Hoffman, M. Brubaker, D. Lee, P. Li, and M. Betancourt. The Stan Math Library: Reverse-mode automatic differentiation in C++. *arXiv preprint arXiv:1509.07164*, 2015.

[6] S. Chandrasekhar. *Radiative transfer*. Dover, 1960.

[7] R. L. Cook and K. E. Torrance. A reflectance model for computer graphics. *ACM Transactions on Graphics (TOG)*, 1(1):7–24, 1982.

[8] A. Delaunoy and E. Prados. Gradient flows for optimizing triangular mesh-based surfaces: Applications to 3d reconstruction problems dealing with visibility. *International journal of computer vision*, 95(2):100–123, 2011.

[9] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159, 2011.

[10] P. Dutre, K. Bala, and P. Bekaert. *Advanced Global Illumination*. A K Peters, Natick, MA, 2006 (2nd edition).

[11] P. Gargallo, E. Prados, and P. Sturm. Minimizing the reprojection error in surface reconstruction from images. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[12] I. Gkioulekas, A. Levin, and T. Zickler. An evaluation of computational imaging techniques for heterogeneous inverse scattering. *European Conference on Computer Vision*, 2016.

[13] I. Gkioulekas, B. Xiao, S. Zhao, E. Adelson, T. Zickler, and K. Bala. Understanding the role of phase function in translucent appearance. *ACM Transactions on Graphics*, 2013.

[14] I. Gkioulekas, S. Zhao, K. Bala, T. Zickler, and A. Levin. Inverse volume rendering with material dictionaries. *ACM Trans. Graph.*, 32(6):162:1–162:13, 2013.

[15] A. Griewank et al. On automatic differentiation. *Mathematical Programming: recent developments and applications*, 6(6):83–107, 1989.

[16] L. Henyey and J. Greenstein. Diffuse radiation in the galaxy. *The Astrophysical Journal*, 93:70–83, 1941.

[17] L. Hosek and A. Wilkie. An analytic model for full spectral sky-dome radiance. *ACM Transactions on Graphics (TOG)*, 31(4):95, 2012.

[18] L. Hošek and A. Wilkie. Adding a solar-radiance function to the Hošek-Wilkie skylight model. *IEEE computer graphics and applications*, 33(3):44–52, 2013.

[19] K. Ikeuchi. Determining surface orientations of specular surfaces by using the photometric stereo method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (6):661–669, 1981.

[20] W. Jakob. Mitsuba renderer, 2010. `http://www.mitsuba-renderer.org`.

[21] J. Kajiya. The rendering equation. *ACM SIGGRAPH Computer Graphics*, 20(4):143–150, 1986.

[22] J. T. Kajiya. The rendering equation. *SIGGRAPH Comput. Graph.*, 20(4):143–150, 1986.

[23] H. Kato, Y. Ushiku, and T. Harada. Neural 3d mesh renderer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3907–3916, 2018.

[24] P. Khungurn, D. Schroeder, S. Zhao, K. Bala, and S. Marschner. Matching real fabrics with micro-appearance models. *ACM Trans. Graph.*, 35(1):1:1–1:26, 2015.

[25] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *ICLR*, 2015.

[26] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *ICLR*, 2014.

[27] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[28] E. Lafortune. *Mathematical models and Monte Carlo algorithms for physically based rendering*. PhD thesis, University of Leuven, 1996.

[29] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[30] A. Levis, Y. Schechner, A. Aides, and A. Davis. Airborne three-dimensional cloud tomography. *IEEE International Conference on Computer Vision*, 2015.

[31] B. Li, C. Shen, Y. Dai, A. van den Hengel, and M. He. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1119–1127, 2015.

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

[33] G. Liu, D. Ceylan, E. Yumer, J. Yang, and J.-M. Lien. Material editing using a physically based rendering network. *ICCV*, 2017.

[34] S. Lombardi and K. Nishino. Reflectance and natural illumination from a single image. *Computer Vision–ECCV 2012*, pages 582–595, 2012.

[35] S. Lombardi and K. Nishino. Reflectance and illumination recovery in the wild. *IEEE transactions on pattern analysis and machine intelligence*, 38(1):129–141, 2016.

[36] M. M. Loper and M. J. Black. Opendr: An approximate differentiable renderer. In *European Conference on Computer Vision*, pages 154–169. Springer, 2014.

[37] S. R. Marschner and D. P. Greenberg. *Inverse rendering for computer graphics*. Cornell University, 1998.

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

[39] X. Mei, H. Ling, and D. W. Jacobs. Illumination recovery from image with cast shadows via sparse representation. *IEEE Transactions on Image Processing*, 20(8):2366–2377, 2011.

[40] A. Meka, M. Maximov, M. Zollhöfer, H.-P. Seidel, C. Richardt, and C. Theobalt. Lime: Live intrinsic material estimation. In *International Conference on Computer Vision and Pattern Recognition*. IEEE, 2018.

[41] Y. Mukaigawa, Y. Yagi, and R. Raskar. Analysis of light transport in scattering media. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 153–160. IEEE, 2010.

[42] S. Narasimhan, M. Gupta, C. Donner, R. Ramamoorthi, S. Nayar, and H. Jensen. Acquiring scattering properties of participating media by dilution. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 1003–1012. ACM, 2006.

[43] T. Narihira, M. Maire, and S. X. Yu. Direct intrinsics: Learning albedo-shading decomposition by convolutional regression. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2992–2992, 2015.

[44] K. Nishino. Directional statistics brdf model. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 476–483. IEEE, 2009.

[45] M. Oren and S. K. Nayar. Generalization of the lambertian model and implications for machine vision. *International Journal of Computer Vision*, 14(3):227–251, 1995.

[46] G. Oxholm and K. Nishino. Multiview shape and reflectance from natural illumination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2155–2162, 2014.

[47] G. Patow and X. Pueyo. A survey of inverse rendering problems. In *Computer graphics forum*, volume 22, pages 663–687. Wiley Online Library, 2003.

[48] G. Patow and X. Pueyo. A survey of inverse surface design from light transport behavior specification. In *Computer Graphics Forum*, volume 24, pages 773–789. Wiley Online Library, 2005.

[49] M. Pharr, W. Jakob, and G. Humphreys. *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2016.

[50] L. B. Rall. Automatic differentiation: Techniques and applications. 1981.

[51] R. Ramamoorthi and P. 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.

[52] K. Rematas, T. Ritschel, M. Fritz, E. Gavves, and T. Tuytelaars. Deep reflectance maps. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4508–4516, 2016.

[53] F. Romeiro and T. Zickler. Blind reflectometry. In *European conference on computer vision*, pages 45–58. Springer, 2010.

[54] S. Sengupta, A. Kanazawa, C. D. Castillo, and D. W. Jacobs. Sfsnet: Learning shape, reflectance and illuminance of faces in the wild. In *International Conference on Computer Vision and Pattern Recognition*, 2018.

[55] J. E. Solem and N. Chr. A geometric formulation of gradient descent for variational problems with moving surfaces. In *Scale-Space*, pages 419–430. Springer, 2005.

[56] I. Sutskever, J. Martens, G. Dahl, and G. Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013.

[57] Y. Tang, R. Salakhutdinov, and G. Hinton. Deep lambertian networks. *arXiv preprint arXiv:1206.6445*, 2012.

[58] D. Terzopoulos, J. Pltt, A. Barr, D. Zeltzer, A. Witkin, and J. Blinn. Physically-based modeling: past, present, and future. *ACM SIGGRAPH Computer Graphics*, 23(5):191–209, 1989.

[59] A. Tewari, M. Zollhöfer, H. Kim, P. Garrido, F. Bernard, P. Pérez, and C. Theobalt. Mofa: Model-based deep convolutional face autoencoder for unsupervised monocular reconstruction. In *The IEEE International Conference on Computer Vision (ICCV)*, volume 2, page 5, 2017.

[60] H.-Y. F. Tung, A. Harley, W. Seto, and K. Fragkiadaki. Adversarial inversion: Inverse graphics with adversarial priors. *ICCV*, 2017.

[61] E. Veach. *Robust monte carlo methods for light transport simulation*. Stanford University Stanford, 1998.

[62] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. 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.

[63] B. Walter, S. R. Marschner, H. Li, and K. E. Torrance. Microfacet models for refraction through rough surfaces. In *Proceedings of the 18th Eurographics conference on Rendering Techniques*, pages 195–206. Eurographics Association, 2007.

[64] X. Wang, D. Fouhey, and A. Gupta. Designing deep networks for surface normal estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 539–547, 2015.

[65] Z. Wang, E. P. Simoncelli, and A. C. Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pages 1398–1402. Ieee, 2003.

[66] G. J. Ward. Measuring and modeling anisotropic reflection. *ACM SIGGRAPH Computer Graphics*, 26(2):265–272, 1992.

[67] T. Weyrich, J. Lawrence, H. P. Lensch, S. Rusinkiewicz, T. Zickler, et al. Principles of appearance acquisition and representation. *Foundations and Trends® in Computer Graphics and Vision*, 4(2):75–191, 2009.

[68] J. Wu, J. B. Tenenbaum, and P. Kohli. Neural scene de-rendering. *CVPR*, 2017.

[69] K.-J. Yoon, E. Prados, and P. Sturm. Joint estimation of shape and reflectance using multiple images with known illumination conditions. *International Journal of Computer Vision*, 86(2):192–210, 2010.

[70] M. D. Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[71] S. Zhao, L. Wu, F. Durand, and R. Ramamoorthi. Downsampling scattering parameters for rendering

anisotropic media. *ACM Trans. Graph.*, 35(6):166:1–166:11, 2016.

[72] T. Zhou, P. Krahenbuhl, and A. A. Efros. Learning data-driven reflectance priors for intrinsic image decomposition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3469–3477, 2015.