Material Editing Using a Physically Based Rendering Network

Guilin Liu[1,3]*   Duygu Ceylan[2]   Ersin Yumer[2]   Jimei Yang[2]   Jyh-Ming Lien[1]
[1]George Mason University   [2]Adobe Research   [3]NVIDIA

Abstract

The ability to edit materials of objects in images is desirable by many content creators. However, this is an extremely challenging task as it requires to disentangle intrinsic physical properties of an image. We propose an end-to-end network architecture that replicates the forward image formation process to accomplish this task. Specifically, given a single image, the network first predicts intrinsic properties, i.e. shape, illumination, and material, which are then provided to a rendering layer. This layer performs in-network image synthesis, thereby enabling the network to understand the physics behind the image formation process. The proposed rendering layer is fully differentiable, supports both diffuse and specular materials, and thus can be applicable in a variety of problem settings. We demonstrate a rich set of visually plausible material editing examples and provide an extensive comparative study.

1. Introduction

One of the main properties of an object that contributes to its appearance, is material. Hence, many designers desire to effortlessly alter materials of objects. In case of 2D design, however, often the designer only has access to a single image of the object. Given one such image, e.g. an image of a fabric sofa, how would you synthesize a new image that depicts the same sofa with a more specular material?

A typical approach to addressing this image-based material editing problem is to first infer the intrinsic properties (e.g. shape, material, illumination) from the image which then enables access to edit them. However, this problem is highly ambiguous since multiple combinations of intrinsic properties may result in the same image. Many prior work has tackled this ambiguity either by assuming at least one of the intrinsic properties is known [13] or devising priors about each of these properties [1]. The recent success of deep learning based methods, on the other hand, has stimulated research to learn such priors directly from the data. A feasible approach is to predict each of the intrinsic proper-

ties independently from an input image, i.e. using individual neural networks to predict normals, material, and lighting. However, the interaction of these properties during the image formation process inspires us to devise a joint prediction framework. An alternative common practice of deep learning methods is to represent the intrinsic properties in a latent feature space [12] where an image can be *decoded* from implicit feature representations. Directly editing such feature representations, however, is not trivial since they do not correspond to any true physical property. On the contrary, the physics of image formation process is well understood and motivates us to replace this *black-box decoder* with a *physical decoder* and thus enables the network to learn the physics of image formation.

We present an end-to-end network architecture that replicates the image formation process in a physically based rendering layer. Our network is composed of prediction modules that infer each of the intrinsic properties from a single image of an object. These predictions are then provided to a rendering layer which re-synthesizes the input image. We define the loss function as a weighted sum of the error over the individual predictions and perceptual error over

Figure 1. We replace the materials of the objects in the input images with the material properties of the objects in the middle column, resulting edited images are shown in the right column.

the synthesized images. We provide comparisons with and without incorporating the perceptual error and show that the combined loss provides significant improvements (see Figure 5 and 6). Due to the lack of a large scale dataset of real images with ground truth normal, material, and lighting annotations, we train our network on a rendering based synthetic dataset. Nevertheless, this network performs reasonable predictions for real images where the space of materials and lighting is much more diverse. We further refine our results on real images with a post-optimization process and show that the network predictions provide a good initialization to this highly non-linear optimization. This paves the road to plausible editing results (see Figure 1).

Our main contributions are:

1. We present an end-to-end network architecture for image-based material editing that encapsulates the forward image formation process in a rendering layer.

2. We present a differentiable rendering layer that supports both diffuse and specular materials and utilizes an environment map to represent a variety of natural illumination conditions. This layer can trivially be integrated into other networks to enable in-network image synthesis and thus boost performance.

2. Related Work

Intrinsic Image Decomposition. A closely related problem to image-based material editing is *intrinsic image decomposition* which aims to infer intrinsic properties, e.g. shape, illumination, and material, from a single image and thus enables access to edit them. The work of Barrow et al. [2] is one of the earliest to formulate this problem and since then several variants have been introduced.

Intrinsic image decomposition is an ill-posed problem since different combinations of shape, illumination, and material may result in the same image. Therefore, an important line of work assumes at least one these unknowns to be given, such as geometry. Patow et al. [20] provides a survey of earlier methods proposed to infer illumination, material, or combination of both for scenes with known geometry. More recent work from Lombardi et al. [13, 14] introduces a Bayesian formulation to infer illumination and material properties from a single image captured under natural illumination. They utilize a material representation based on Bidirectional Reflectance Distribution Functions (BRDFs) and thus can handle a wide range of diffuse and specular materials. Approaches that aim to infer all three intrinsic properties [1], on the other hand, often make simplified prior assumptions such as diffuse materials and low-frequency lighting to reduce the complexity of the problem.

An important sub-class of intrinsic image decomposition is *shape from shading (ShS)* where the goal is to reconstruct accurate geometry from shading cues [28]. Many ShS approaches, however, assume prior knowledge about the material properties [9] or coarse geometry [27] to be given.

As opposed to these approaches, we assume no prior knowledge about any of the intrinsic properties to be given and handle both diffuse and specular materials under natural illumination. Instead of making assumptions, we aim to infer priors directly from data via a learning based approach.

Material Editing. Some previous methods treat material editing as an image filtering problem without performing explicit intrinsic image decomposition. Khan et al. [10] utilize simple heuristics to infer approximate shape and illumination from an image and utilize this knowledge to perform material editing. Boyadzhiev et al. [3] introduce several image filters to change properties such as shininess and glossiness. While these approaches achieve photo-realistic results they can provide limited editing scenarios without explicit knowledge of the intrinsic properties.

Material Modeling via Learning. With the recent success of learning based methods, specifically deep learning, several data-driven solutions have been proposed to infer intrinsic properties from an image. Tang et al. [25] introduce *deep lambertian networks* to infer diffuse material properties, a single point light direction, and an orientation map from a single image. They utilize Gaussian Restricted Boltzmann Machines to model surface albedo and orientation. Richter et al. [22] use random forests to extract surface patches from a database to infer the shape of an object with diffuse and uniform albedo. Narihira et al. [17] predict relative lightness of two image patches by training a classifier on features extracted by deep networks. Similarly, Zhou et al. [29] use a convolutional neural network (CNN) to predict relative material properties of two pixels in an image and then perform a constrained optimization to solve for the albedo of the entire image. Narihira et al. [24] propose a CNN architecture to directly predict albedo and shading from an image. Kulkarni et al. [12] use variational auto-encoders to disentangle viewpoint, illumination, and other intrinsic components (e.g. shape, texture) in a single image. One of the limitation of these methods is the ability to handle diffuse materials only. The recent work of Rematas et al. [21] predict the combination of material and illumination from a single image handling both diffuse and specular materials. In a follow-up work [6], they propose two independent network architectures to further disentangle material and illumination from such a combined representation. However, any error that occurs when inferring the combined representation is automatically propagated to the second step. In contrast, we provide an end-to-end network architecture that performs this disentangling in one pass. Last but not least, several recent work [23, 7] uses neural networks to estimate per-pixel intrinsic properties from a given image. However, these estimations are aligned with the input image, thus, it is not easy transfer these properties across images of different objects as our method does.
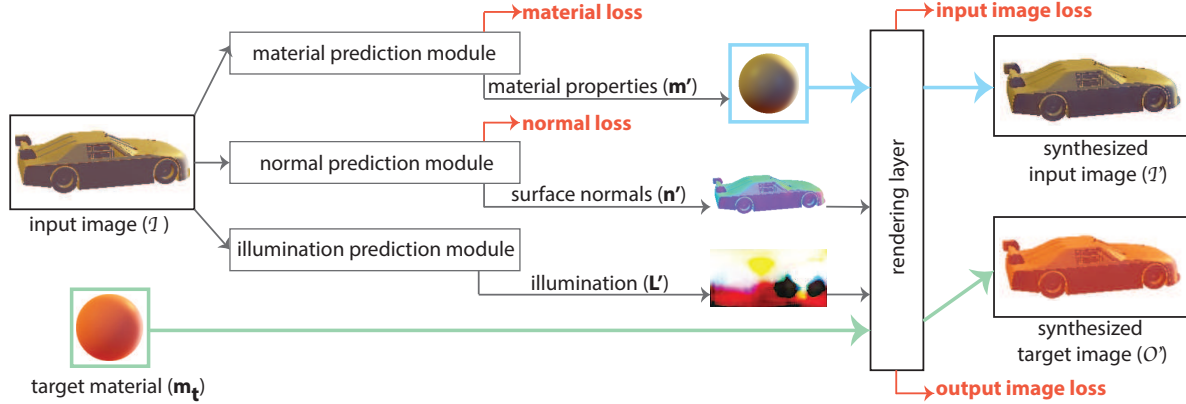
Figure 2. Given a single image of an object, I, we propose a network architecture to predict material ($\mathbf{m'}$), surface normals ($\mathbf{n'}$), and illumination ($\mathbf{L'}$). These predictions are provided to a rendering layer which re-synthesizes the input image ($\mathbf{I'}$). In addition, a desired target material, $\mathbf{m_t}$, is passed to the rendering layer along with $\mathbf{n'}$ and $\mathbf{L'}$ to synthesize a target image $\mathcal{O'}$ which depicts the object with the target material from the same viewpoint and under the same illumination. By defining a joint loss that evaluates both the synthesized images and the individual predictions, we perform robust image decomposition and thus enable material editing applications.

3. Approach

Overview. We present an end-to-end network architecture for image-based material editing. The input is $\mathcal{I}$, a single image of an object s with material $\mathbf{m}$ captured under illumination $\mathbf{L}$. We assume the object is masked out in the image. Given a desired target material definition $\mathbf{m_t}$, the goal is to synthesize an output image $\mathcal{O}$ that depicts s from the same viewpoint with material $\mathbf{m_t}$ and illuminated by $\mathbf{L}$.

While inferring illumination, material, and shape from a single image is an ill-posed problem, once these parameters are known, the forward process of image synthesis, i.e. *rendering* is well defined. We propose a network architecture that encapsulates both this inverse decomposition and the forward rendering processes as shown in Figure 2. The first part of the network aims to infer illumination ($\mathbf{L'}$), material ($\mathbf{m'}$), and 3D shape (represented as surface normals, $\mathbf{n'}$) from $\mathcal{I}$ via three prediction modules. The output of these modules is then provided to a *rendering layer* to synthesize $\mathcal{I'}$. In addition, the target material $\mathbf{m_t}$ is passed to the rendering layer along with predicted $\mathbf{L'}$ and $\mathbf{n'}$ to synthesize an output image O'. We define a joint loss that evaluates both the outputs of the rendering layer and the predictions of two of the individual modules. We show that this joint loss significantly improves the accuracy of the inverse decomposition and enables compelling material editing results.

We first introduce our render layer (Sec. 3.1), prediction modules providing data to it (Sec. 3.2), and the training procedure (Sec. 3.3). Then, we discuss how objects composed of multiple materials are handled (Sec. 3.4).

3.1. Rendering Layer

The color of each pixel in an image depends on how the corresponding object surface point reflects and emits in-coming light along the viewing direction. Our rendering layer aims to replicate this process as accurately as possible. Compared to previously proposed differentiable renderers [15], the main advantage of our rendering layer is to model surface material properties based on BRDFs to handle both diffuse and specular materials. We also utilize environment maps which provide great flexibility in representing a wide range of illumination conditions. We note that our rendering layer also makes several moderate assumptions. We assume the image is formed under translation-invariant natural illumination (i.e. incoming light depends only on the direction). We also assume there is no emission and omit complex light interactions like inter-reflections and subsurface scattering.

Under these conditions, we model the image formation process mathematically similar to Lombardi et al [13]. Given per-pixel surface normals $\mathbf{n}$ (in camera coordinates), material properties $\mathbf{m}$ and illumination $\mathbf{L}$, the outgoing light intensity for each pixel p in image $\mathcal{I}$ can be written as an integration over all incoming light directions ω_i:

$$\mathcal{I}_p(\mathbf{n_p}, \mathbf{m}, \mathbf{L}) = \int f(\omega_i, \omega_o, \mathbf{m}) \mathbf{L}(\omega_i) \max(0, \mathbf{n_p} \cdot \omega_i) d\omega_i,$$

(1)

where $\mathbf{L}(\omega_i)$ defines the intensity of the incoming light and $f(\omega_i, \omega_o, \mathbf{m})$ defines how this light is reflected along the outgoing light direction ω_o based on the material properties $\mathbf{m}$. In order to make this formulation differentiable, we substitute the integral with a sum over a discrete set of incoming light directions defined by the illumination $\mathbf{L}$:

$$\mathcal{I}_p(\mathbf{n_p}, \mathbf{m}, \mathbf{L}) = \sum_{\mathbf{L}} f(\omega_i, \omega_o, \mathbf{m}) \mathbf{L}(\omega_i) \max(0, \mathbf{n_p} \cdot \omega_i) d\omega_i.$$

(2)

We now describe how we represent each property and re-

fer to the supplementary material about the details of the forward and back propagation on this rendering layer.

Surface normals (n). $\mathbf{n}$ is represented by a 3-channel image, same size as the input image, where each pixel p encodes the corresponding per-pixel normal $\mathbf{n_p}$.

Illumination (L). We represent illumination with an HDR environment map of dimension 64×128. Each pixel coordinate in this image can be mapped to spherical coordinates and thus corresponds to an incoming light direction ω_i in Equation 2. The pixel value stores the intensity of the light coming from this direction.

Material (m). We define $f(\omega_i, \omega_o, \mathbf{m})$ based on BRDFs [18] which provide a physically correct description of pointwise light reflection both for diffuse and specular surfaces. Non-parametric models [16] aim to capture the full spectrum of BRDFs via lookup tables that encode the ratio of the reflected radiance to the incident radiance given incoming and outgoing light directions (ω_i, ω_o). Although such lookup tables achieve highly realistic results, they are computationally expensive to store and not differentiable. Among the various parametric representations, we adopt the Directional Statistics BRDF (DSBRDF) model [19] which is shown to accurately model a wide variety of measured BRDFs. This model represents each BRDF as a combination of hemispherical exponential power distributions and the number of parameters depends on the number of distributions utilized. Our experiments show that utilizing 3 distributions provides accurate approximations resulting in 108 parameters per material definition in total. We refer the reader to the original work [19] for more details.

3.2. Prediction Modules

We utilize three prediction modules to infer surface normals, material properties, and illumination. The input to each module is the 256×256 input image $\mathcal{I}$. We refer to the supplementary material for the detailed network architecture of each module.

Normal prediction. The normal prediction module follows the same spirit as the recent work of Eigen et al. [5]. The main difference is that we predict a normal map that is equal in size to the input image by utilizing a 4-scale-submodule network as opposed to the originally proposed 3-scale-submodule network. The fourth submodule consists of 2 convolutional layers where both input and output size is equal to the size of the input image. We utilize a normalization layer to predict surface normals with unit length.

Illumination prediction. Illumination prediction module is composed of 7 convolutional layers where the output of each such layer is half the size of its input. The convolutional layers are followed by 2 fully connected layers and a sequence of deconvolutional layers to generate an environment map of size 64×128. Each convolutional and fully connected layer is accompanied by a rectifier linear unit.

Fully connected layers are also followed by dropout layers.

Material prediction. Material prediction module is composed of 7 convolutional layers where the output of each such layer is half the size of its input. The convolutional layers are followed by 3 fully connected layers and a tanh layer. Each convolutional and fully connected layer is accompanied by a rectifier linear unit. Fully connected layers are also followed by dropout layers. We note that since each of the 108 material parameters are defined at different scales, we normalize each to the range $[-0.95, 0.95]$ and remap to their original scales after prediction.

3.3. Training.

We train the proposed network architecture by defining a joint loss function that evaluates both the predictions of the individual modules and the images synthesized by the rendering layer. Specifically, we define a L2 normal loss:

$$l_{normal} = \sum_p (\mathbf{n_p} - \mathbf{n'_p})^2, \qquad (3)$$

where $\mathbf{n_p}$ and $\mathbf{n'_p}$ denote the ground-truth and predicted surface normals for each pixel p respectively. Since both ground-truth and predicted normals are unit length, this loss is equivalent to cosine similarity used by Eigen et al. [5].

We define the material loss as the L2 norm between the ground truth ($\mathbf{m}$) and predicted material parameters ($\mathbf{m'}$):

$$l_{material} = (\mathbf{m} - \mathbf{m'})^2 \qquad (4)$$

We also utilize a perceptual loss, $l_{perceptual}$, to evaluate the difference between the synthesized $\{\mathcal{I'}, \mathcal{O'}\}$ and ground truth $\{\mathcal{I}, \mathcal{O}\}$ images, which helps to recover normal details not captured by the L2 normal loss. We use the pre-trained *vgg16* network to measure $l_{perceptual}$ as proposed by Johnson et al. [8].

We define the final loss, l, as a weighted combination of these individual loss functions:

$$l = w_n l_{normal} + w_m l_{material} + w_p l_{perceptual}, \quad (5)$$

where we empirically set $w_n{=}1 \times 10^4$, $w_m{=}1 \times 10^3$, $w_p{=}1$.

Note that we do not define a loss between ground truth and predicted illumination due to the limited amount of available public HDR environment maps for training. Each pixel in the spherical environment map corresponds to a light direction. Depending on the viewing direction and surface normals, the contribution of each light direction to the final rendered image will vary. We leave it as future work to design a robust loss function that takes this non-uniformity into account.

3.4. Extension to Multi-Material Case

A typical use case scenario for image-based material editing is where a user denotes the region of interest of an

Figure 3. For each input image, we synthesize an output image with a given target material definition. We provide the ground truth (GT) target images for reference. The left (middle) column shows cases where the target material is more (less) specular than the input material. We also provide examples where both the input and the target material are specular in the right column.

object and a desired target material that should be applied to this region. Our approach trivially supports such use cases. Given a segmentation mask denoting regions of uniform material, we perform material prediction for each region while predicting a single normal and illumination map for the entire image. All these predictions are then provided to the rendering layer. Target materials are defined for each region of interest separately. The rest of the training and testing process remains unchanged.

3.5. Refinement with Post-optimization

Due to the lack of a large scale dataset of real images with ground truth normal, material, and illumination annotations, we train our network with synthetic renderings. Although large 3D shape repositories [4] provide sufficient shape variation, the network does not see a large spectrum of material and illumination properties captured in real images. While increasing the variation in the synthetic data could reduce the discrepancy between the training and test sets, the gap is almost always there theoretically. Thus, we refine our network predictions on real images with a post-optimization. Specifically, using $\mathbf{n}', \mathbf{m}', \mathbf{L}'$, the network predictions of surface normal, material, and illumination for an input image $\mathcal{I}$ as initialization, we optimize for $\mathbf{n}^*, \mathbf{m}^*, \mathbf{L}^*$ which minimize the following energy function:

$$\operatorname*{argmin}_{\mathbf{n}^*,\mathbf{m}^*,\mathbf{L}^*} \parallel \mathcal{I}^* - \mathcal{I} \parallel^2 + a \parallel \mathbf{n}^* - \mathbf{n}' \parallel^2 + b \parallel \mathbf{L}^* - \mathbf{L}' \parallel^2 .$$
(6)

$\mathcal{I}^*$ is the image formed by $\mathbf{n}^*, \mathbf{m}^*, \mathbf{L}^*$ and the first term penalizes the image reconstruction loss. The remaining regularization terms penalize the difference between the network predictions of normal and illumination and their optimized values. We experimentally set a=1, b=10 as the relative weighting of these regularization terms. We use L-BFGS to solve Equation 6 in an alternative scheme where only one intrinsic property is updated at each iteration.

4. Evaluation

In this section we provide quantitative and qualitative evaluations of our method and comparisons to other work.

4.1. Datasets and Training

We train the framework with a large amount of synthetic data generated for *car, chair, sofa*, and *monitor* categories. We utilize a total of 280 3D models (130 cars, 50 chairs, 50 sofas and 50 monitors) obtained from ShapeNet [4]. For materials, we use BRDF measurements corresponding to 80 different materials provided in the MERL database [16]. For illumination, we download 10 free HDR environment maps from the Internet and use random rotations to augment them.

Data generation. For each 3D model, material, and illumination combination we render 5 random views from a fixed set of pre-sampled 24 viewpoints around the object with fixed elevation. We split the data such that no shape and material is shared between training and test sets. Specifically, we use 80 cars for pretraining, and 40 shapes per each category for joint category finetuning, which leaves 10 shapes per category for testing. Out of the 80 materials, we leave 20 for testing and use the rest in pre-training and training. This split allows us to generate $240K$ pretraining instances, and $480K$ multi-category finetuning instances. In total, the network is trained with over $720K$ unique material-shape-light configurations.

Training procedure. We initialize the weights from a uniform distribution: $[-0.08, 0.08]$. Normal and material modules are pre-trained using L2 loss for a few iterations and then trained jointly with illumination network. We utilize Adam [11] optimizer using stochastic gradient descent. Similarly, we first used momentum parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and a learning rate of 0.0001. Later we reduced the learning rate to be 1×10^{-6} and 1×10^{-8}. We also reduced β_1 to 0.5 for a more stable training.

4.2. Evaluation on Synthetic Data

To test our network, we randomly select an image corresponding to shape s, environment map $\mathbf{L}$, and material $\mathbf{m}$ from our test set as input. Given a target material $\mathbf{m_t}$, an output image depicting s with material $\mathbf{m_t}$ and illuminated by $\mathbf{L}$ is synthesized. We compare this synthesized output to ground truth using two metrics. While the L2 met-
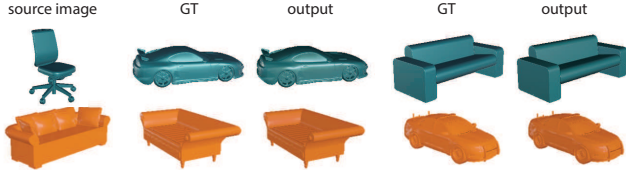
Figure 4. We synthesize an image of an object under some desired illumination with material properties predicted from a source image. We provide the ground truth (GT) for reference.

Figure 5. For each example we show the ground truth image as well as the renderings obtained by utilizing the material coefficients predicted by the standalone material prediction module and the combined approach which also uses the rendering layer.

ric measures the average pixel-wise error, SSIM metric[26] measures the structural difference. We compute the L2 error on tone mapped images where each color channel is in the range $[0, 255]$ and define per-pixel error as the average of the squared difference of three color channels. We note that lower (higher) numbers are better for the L2 (SSIM) metric. We provide quantitative results in Table 3, last two columns. We note that our network is trained for all categories jointly. Figure 3 provides visual results demonstrating that our approach can successfully synthesize specularities when replacing the input material with a more specular target material. Similarly, specularities in the input image are successfully removed when replacing the input material with a more diffuse one.

In Figure 4, we show examples where material properties predicted from a source image are transferred to a different object under some desired illumination. Such material transfers alleviates the need to define target materials explicitly, instead each image becomes an exemplar.

4.3. Evaluation of the Rendering Layer

We evaluate the effectiveness of utilizing the perceptual loss on the images synthesized by the rendering layer as opposed to independent predictions. We note that for this evaluation, we directly use the pure network predictions without post-optimization.

Accuracy of Material Coefficients. For this evaluation, given an input image $\mathcal{I}$, we provide ground truth normals ($\mathbf{n}$) and illumination ($\mathbf{L}$) and use our network to predict only material. We also train the material prediction module as a standalone network. While both the standalone module and our network use the L2 loss on material coefficients in spirit similar to the recent work of Georgoulis et al. [6], the latter combines this with the perceptual loss on $\mathcal{I}'$. Note that during training, the material parameters have been normalized to the range [-0.95,0.95] to reduce the scale issues for L2 loss; while in Table 1, we remap to the original scales. 60 materials are used for training and 20 for testing. We provide qualitative and quantitative results in Figure 5 and Table 1 respectively. The results demonstrate that the L2 loss is not sufficient in capturing the physical material properties, a small L2 error may result in big visual discrepancies in the rendering output. Incorporation of the rendering layer

resolves this problem by treating the material coefficients in accordance with their true physical meaning.

| | material | rendering | |
	L2	L2	SSIM
standalone	4.1038	544.1	0.9297
combined	4.8144	355.5	0.9517

Table 1. We show the accuracy of the predicted material coefficients and the images rendered using these together with ground truth normals and illumination. We provide results of the material prediction module trained standalone vs with the rendering layer.

Accuracy of Surface Normals. For this evaluation, given an input image $\mathcal{I}$, we provide ground truth material ($\mathbf{m}$) and illumination ($\mathbf{L}$) and use our network to predict only surface normals. We use the *car* category for the evaluation. The predicted surface normals along with $\mathbf{m}$ and $\mathbf{L}$ are used to synthesize $\mathcal{I}'$. We also train the normal prediction module as a standalone network. While both networks use the L2 loss on the surface normals, our network combines this with the perceptual loss on $\mathcal{I}'$.

We provide qualitative and quantitative results in Figure 6 and Table 2 respectively. We observe that L2 loss helps to get the orientation of the surface normals correct, and thus results in small errors, but produces blurry output. Adding the perceptual loss helps to preserve sharp features resulting in visually more plausible results. We also note that, although commonly used, L2 and SSIM metrics are not fully correlated with human perception. Incorporation of the rendering layer results in slim improvements in quantitative numbers (Table 1 and 2) but significantly better appearance results as shown in Figure 5 and 6.
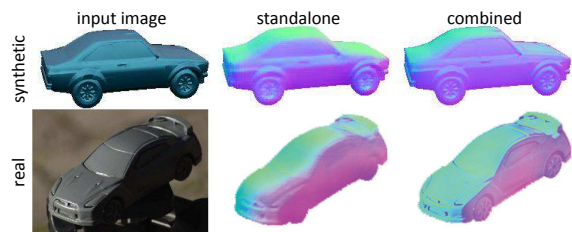

Figure 6. For a synthetic and a real input image, we provide the surface normals predicted by the normal prediction module when trained standalone vs with the rendering layer.
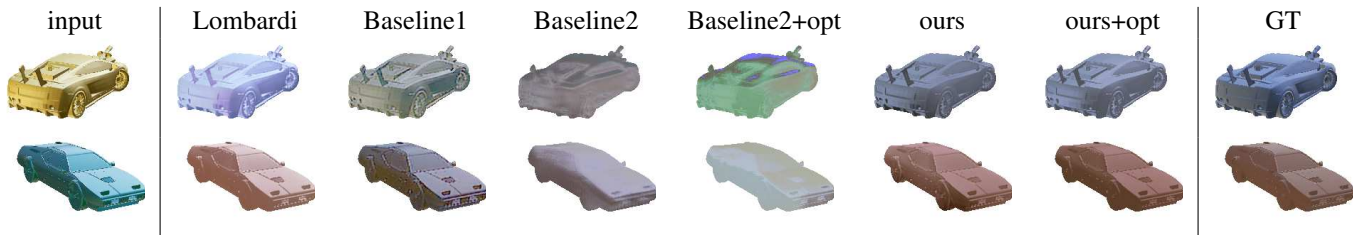
Figure 7. For each example we show the input and the ground truth target image (GT). We provide the results obtained by the method of Lombardi et al. [13], two baseline methods, and our approach.

	normals	rendering	
	cosine	L2	SSIM
standalone	0.9105	313.2058	0.9441
combined	0.9050	238.4619	0.9625

Table 2. We show the accuracy of the predicted surface normals and the images rendered using these together with ground truth material and illumination. We provide the results of the normal prediction module trained standalone vs with the rendering layer.

4.4. Multi-Material Examples

We show examples for objects composed of multiple materials. For these examples, we manually segment 100 chair models to two semantic regions where each region gets its own material assignment. We generate renderings from 5 different viewpoints with random illumination and materials assignments. We fine-tune the model trained on single-material examples with these additional renderings. We utilize 80 chairs for fine-tuning and 20 for testing. In Figure 8, we show examples of editing each region with a target material. We also quantitatively evaluate these results with respect to ground truth over 4000 examples. We achieve an L2 error of 1272.1 and SSIM index of 0.9362. No post-optimization is used for the experiments.
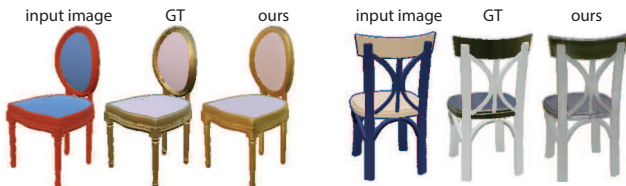

Figure 8. Given an image of a multi-material object, we show how different target material assignments are realized for different parts. We provide ground target (GT) images for reference.

4.5. Comparisons

We compare our method with several baseline methods and previous approaches which we briefly discuss below and refer to the supplementary material for more details.

Lombardi et al. [13] We compare our method to the approach of Lombardi et al. [13] which uses certain priors to optimize for illumination and material properties in a single image of an object with known geometry. We provide the ground truth surface normals to this method and predict the material and illumination coefficients. We render the target image given the target material, ground truth normals, and the predicted illumination.

Baseline 1. Our first baseline is inspired by the work of Kulkarni et al. [12] and is based on an encoder-decoder network. The encoder maps the input image to a feature representation, $f = (f_m, f_o)$, with one part corresponding to the material properties (f_m) and the remaining part corresponding to other factors (f_o) (i.e. normals and illumination). This disentangling is ensured by passing f_m as input to a regression network which is trained to predict the input material coefficients. f_m is then replaced by the feature representation of the target material. The decoder utilizes the updated feature representation to synthesize the target image. We also provide skip connections from the input image to the decoder to avoid blurry output. Similar to our approach, *baseline 1* is trained with the perceptual loss between the synthesized and ground truth target image and the L2 loss on the estimated input material coefficients.

Baseline 2. For the second baseline, given the input image we train three individual networks to predict the surface normals, material, and illumination separately. All networks are trained with the L2 loss. We then render the target image using the predicted surface normals, illumination, and the target material.

For this comparison we train our method and both of the baselines on the same training and testing set. We refine the output of baseline 2 and our network with the post-optimization described in Section 3.5. We use 500 images sampled from the testing set as input and generate a new image with a target material from the test set. We provide quantitative (see Table 3) and visual results (see Figure 7). For baseline 2, the use of L2 loss only results in blurry normal prediction, hence the blurry target images. While baseline 1 addresses this issue by use of skip connections, implicit feature representations fail to capture the true intrinsic properties and result in a large appearance difference with respect to ground truth. The approach of Lombardi et al. [13] performs non-convex optimization over the ren-

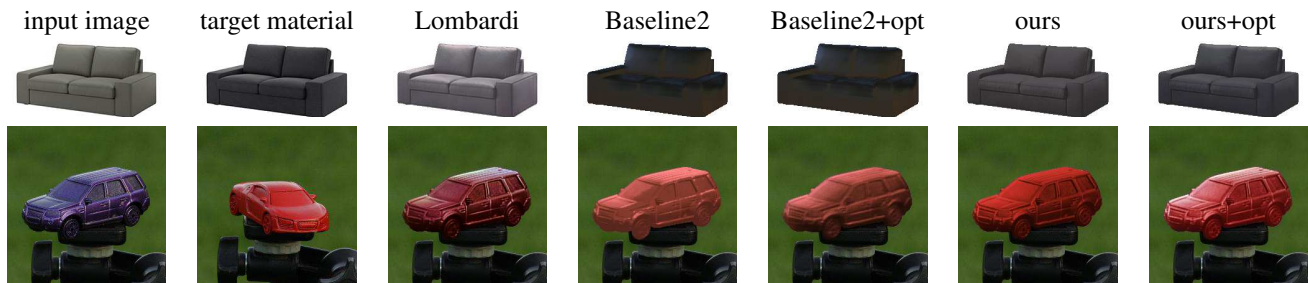| input image | target material | Lombardi | Baseline2 | Baseline2+opt | ours | ours+opt |

Figure 9. We transfer the target material to the input image using the method of Lombardi et al. [13], baseline 2, baseline 2 with post-optimization, our network, and our network with post-optimization.

| | Lombardi | Baseline1 | Baseline2 | | ours | |
			no opt	opt	no opt	opt
L2	802.7	1076.7	886.9	1273.7	805.2	530.9
SSIM	0.9416	0.9173	0.9256	0.9159	0.9408	0.9557

Table 3. We evaluate the accuracy of material transfer results on synthetic data for Lombardi et al. [13], baseline 1, baseline 2 with and without post-optimization, and our method with and without post-optimization.

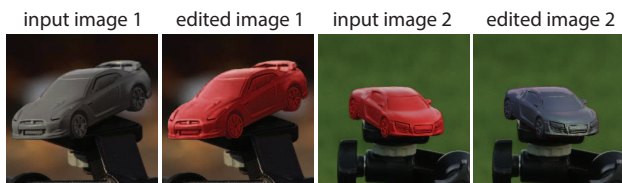| input image 1 | edited image 1 | input image 2 | edited image 2 |

Figure 10. Given two input images, we synthesize new images by swapping the material properties of the objects.

dering equation which is likely to get stuck in local minimum without a good initialization. The individual predictions from baseline 2 are quite far from the global minimum or any reasonable local minimum. Thus, we observe no improvement with post-optimization, in some cases the optimization in fact gets stuck in a worse local minimum. Our initial network output performs in par with [13] with no assumption on known surface normals. Refining these predictions with the post-optimization outperforms all other methods significantly.

4.6. Evaluation on Real Images

We provide visual comparisons of our method with the method of Lombardi et al. [13] and baseline 2 on real product images[1] downloaded from the internet in Figure 9. Since the method of Lombardi et al. [13] assumes surface normals to be known, we provide the normal prediction of our network to their optimization. Additionally, we refine the results of both baseline 2 (i.e. individual prediction of normal, material, and illumination) and our network with the

---

[1]For real examples we finetune our network using all materials in our dataset to better generalize to unseen materials.

post-optimization of Section 3.5. Even though our network has been trained only on synthetic data with a limited set of environment maps and materials, the raw network results are promising and provide a good initialization to the post-optimization. Independent predictions, on the other hand, result in the post-optimization to get stuck in a local minimum which does not yield as visually plausible results.

Finally, in Figure 10, we provide material transfer examples on some real images provided by the SMASHINg challenge dataset [21] with the combination of our network and the post optimization. We refer to the supplementary material for more examples and comparisons.

5. Conclusion and Future Work

We propose an end-to-end network architecture for image-based material editing that encapsulates the image formation process in a rendering layer. We demonstrate various plausible material editing results both for synthetic and real data. One of the limitations of our work is the fact that lighting space is learned with a relatively small dataset, because of the lack of data. This may result in imperfect decomposition, specifically some lighting effects being predicted to be part of the surface normals. For multi-material cases, estimating material for a small segment independently is difficult. For such scenarios, extending the network to perform per-pixel material predictions will be a promising direction. Incorporation of advanced light interactions such as subsurface scattering and inter-reflections in the rendering layer is crucial to generalize the method to scenes with multiple objects and real images. We expect the performance boost obtained by the encapsulation of the rendering layer to stimulate further research in designing neural networks that can replicate physical processes.

References

[1] J. T. Barron and J. Malik. Shape, illumination, and reflectance from shading. *IEEE PAMI*, 2015. 1, 2

[2] H. G. Barrow and J. M. Tenenbaum. Recovering intrinsic scene characteristics from images. Technical Report 157, AI Center, SRI International, 333 Ravenswood Ave., Menlo Park, CA 94025, Apr 1978. 2

[3] I. Boyadzhiev, K. Bala, S. Paris, and E. Adelson. Band-sifting decomposition for image-based material editing. *ACM TOG*, 34(5):163:1–163:16, Nov. 2015. 2

[4] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 5

[5] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *IEEE ICCV*, pages 2650–2658, 2015. 4

[6] S. Georgoulis, K. Rematas, T. Ritschel, M. Fritz, L. V. Gool, and T. Tuytelaars. Delight-net: Decomposing reflectance maps into specular materials and natural illumination. arxiv:1603.08240 [cs.cv], 2016. 2, 6

[7] C. Innamorati, T. Ritschel, T. Weyrich, and N. J. Mitra. Decomposing single images for layered photo retouching. In *Computer Graphics Forum*, volume 36, pages 15–25. Wiley Online Library, 2017. 2

[8] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *IEEE ECCV*, 2016. 4

[9] M. K. Johnson and E. H. Adelson. Shape estimation in natural illumination. In *Computer Vision and Pattern Recognition (CVPR)*, pages 2553–2560, 2011. 2

[10] E. A. Khan, E. Reinhard, R. W. Fleming, and H. H. Bülthoff. Image-based material editing. In *ACM SIGGRAPH*, pages 654–663, 2006. 2

[11] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[12] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. Tenenbaum. Deep convolutional inverse graphics network. In *NIPS*, pages 2539–2547. 2015. 1, 2, 7

[13] S. Lombardi and K. Nishino. Reflectance and natural illumination from a single image. In *IEEE ECCV*, pages 582–595. Springer, 2012. 1, 2, 3, 7, 8

[14] S. Lombardi and K. Nishino. Reflectance and illumination recovery in the wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(1):129–141, Jan 2015. 2

[15] M. M. Loper and M. J. Black. OpenDR: An approximate differentiable renderer. In *Computer Vision – ECCV 2014*, volume 8695 of *Lecture Notes in Computer Science*, pages 154–169. Springer International Publishing, Sept. 2014. 3

[16] W. Matusik, H. Pfister, M. Brand, and L. McMillan. A data-driven reflectance model. *ACM Transactions on Graphics*, 22(3):759–769, July 2003. 4, 5

[17] T. Narihira, M. Maire, and S. X. Yu. Learning lightness from human judgement on relative reflectance. In *IEEE CVPR*, June 2015. 2

[18] F. E. Nicodemus, J. C. Richmond, J. J. Hsia, I. W. Ginsberg, and T. Limperis. Radiometry. chapter Geometrical Considerations and Nomenclature for Reflectance, pages 94–145. Jones and Bartlett Publishers, Inc., USA, 1992. 4

[19] K. Nishino. Directional statistics brdf model. In *2009 IEEE 12th International Conference on Computer Vision*, pages 476–483. IEEE, 2009. 4

[20] G. Patow and X. Pueyo. A survey of inverse rendering problems. In *CGF*, volume 22, pages 663–687. Wiley Online Library, 2003. 2

[21] K. Rematas, T. Ritschel, M. Fritz, E. Gavves, and T. Tuytelaars. Deep reflectance maps. In *CVPR*, 2016. 2, 8

[22] S. R. Richter and S. Roth. Discriminative shape from shading in uncalibrated illumination. In *IEEE CVPR*, pages 1128–1136, June 2015. 2

[23] J. Shi, Y. Dong, H. Su, and S. X. Yu. Learning non-lambertian object intrinsics across shapenet categories. *arXiv preprint arXiv:1612.08510*, 2016. 2

[24] M. M. Takuya Narihira and S. X. Yu. Direct intrinsics: Learning albedo-shading decomposition by convolutional regression. In *International Conference on Computer Vision (ICCV)*, 2015. 2

[25] Y. Tang, R. Salakhutdinov, and G. Hinton. Deep lambertian networks. In *ICML*, 2012. 2

[26] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. on Image Proc.*, 13(4):600–612, 2004. 6

[27] L. F. Yu, S. K. Yeung, Y. W. Tai, and S. Lin. Shading-based shape refinement of rgb-d images. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1415–1422, June 2013. 2

[28] R. Zhang, P.-S. Tsai, J. E. Cryer, and M. Shah. Shape from shading: A survey. *IEEE PAMI*, 21(8):690–706, 1999. 2

[29] T. Zhou, P. Krhenbhl, and A. A. Efros. Learning data-driven reflectance priors for intrinsic image decomposition. In *IEEE ICCV*, pages 3469–3477, 2015. 2