Intrinsic Autoencoders for Joint Neural Rendering and Intrinsic Image Decomposition

Hassan Abu Alhaija*,[1] Siva Karthik Mustikovela*,[1] Justus Thies[2]
Varun Jampani[3] Matthias Nießner[2] Andreas Geiger[4,5] Carsten Rother[1]

[1]Heidelberg University [2]Technical University Munich [3]NVIDIA
[4]Max Planck Institute for Intelligent Systems, Tübingen
[5]University of Tübingen

hassanhaija@gmail.com, siva.mustikovela@iwr.uni-heidelberg.de

Abstract. Neural rendering techniques promise efficient photo-realistic image synthesis while at the same time providing rich control over scene parameters by learning the physical image formation process. While several supervised methods have been proposed for this task, acquiring a dataset of images with accurately aligned 3D models is very difficult. The main contribution of this work is to lift this restriction by training a neural rendering algorithm from unpaired data. More specifically, we propose an autoencoder for joint generation of realistic images from synthetic 3D models while simultaneously decomposing real images into their intrinsic shape and appearance properties. In contrast to a traditional graphics pipeline, our approach does not require to specify all scene properties, such as material parameters and lighting by hand. Instead, we learn photo-realistic deferred rendering from a small set of 3D models and a larger set of unaligned real images, both of which are easy to acquire in practice. Simultaneously, we obtain accurate intrinsic decompositions of real images while not requiring paired ground truth. Our experiments confirm that a joint treatment of rendering and decomposition is indeed beneficial and that our approach outperforms state-of-the-art image-to-image translation baselines both qualitatively and quantitatively.

1 Introduction

State-of-the-art sampling-based rendering engines (e.g., Mitsuba[21]) are able to generate photo-realistic images of virtual objects which are nearly indistinguishable from real-world photographs. However, this is not an easy task to accomplish since all intrinsic physical aspects of the virtual object must be accurately modeled, such as accurate 3D geometry, detailed textures and physically-based materials. While some of these intrinsics are abundant on the internet, such as the geometry of 3D objects (e.g. Turbosquid and 3D Warehouse), others are hard to obtain, such as high-quality materials – ideally in the form of a highly-accurate spatially-varying BRDF. In addition, sophisticated and slow rendering

* Equal contribution

algorithms with many tunable parameters (lighting, environment map, camera model, post-processing) are required for turning 3D content into photo-realistic images. These parameters are often tuned individually with each rendered image, making it hard to create a large and diverse set of rendered images. On the other hand, obtaining a large number of real images which capture the complex interaction of light with scene geometry and surface properties is easy. This makes the idea of learning neural image synthesis from real images very attractive.

Several works on conditional image generation [35,38,20,8] have exploited paired datasets of real images with semantic information, including semantic segmentation [35,8] and body part labels [26] for training realistic image synthesis models. However, such sparse inputs only allow for a low level of control over the generated image. This limits the applicability of these methods, e.g., in virtual reality or video game simulations where precise control over the output is essential. Training a conditional image generation model from richer control inputs would require a large dataset of paired real images with pixel-accurately aligned intrinsic properties such as 3D structure, textures, materials and reflections. Unfortunately, obtaining such a dataset is hard in practice.

In this work, our goal is to take a step towards learning a highly controllable realistic image synthesis model without requiring real world images with aligned 3D models. Our key insight is that learning the inverse task of intrinsic decomposition is helpful for learning image synthesis from real images and vice-versa. We therefore train both, the forward rendering process and the reverse intrinsic decomposition process, jointly using a single objective as illustrated in Fig. 1. Inspired by recent results in unpaired image-to-image translation [18,30,43], we train our model using an small set of synthetic 3D models of a particular object category as well as a large unpaired dataset of real images of the same category.

Towards this goal, we exploit a technique from real-time rendering called *Deferred Rendering* which splits the rendering process into two stages and thus improves efficiency. In the first stage, the geometry of the scene along with its textures and material properties are projected onto a 2D pixel grid, resulting in a set of 2D intrinsic images which capture the geometry and appearance of the object. This step is efficient since it does not require physically accurate path tracing but relies on simple rendering operations. In the second "deferred" stage, lighting, shading and textural details are added to form the final rendered image. Our goal is to replace this second deferred stage of the rendering process with a neural network which we call *Deferred Neural Rendering* (DNR) network. To ensure that the input information is represented in the output image, we decompose it back into its intrinsics using a second Intrisic Image Decomposition (IID) network. However, we found that using this cycle alone leads to overfitting, especially in the IID network. This is likely because the IID learns to decompose images generated by the NDR and therefore can learn to decode information hidden in the output of the NDR rather than learning the harder process of intrinsic decomposition from visual appearance. To improve the IID network, we introduce a second *Decomposition cycle* in which we train the IID network to decompose real images.

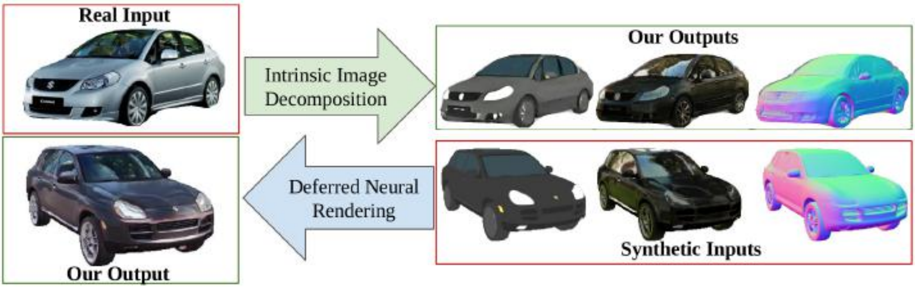


Fig. 1: **Joint Deferred Rendering and Intrinsic Image Decomposition.** At training time, our model exploits normals, albedo and reflections from a small set of 3D models as well as a large set of *unpaired* RGB images of the same object category. Our model solves two tasks simultaneously: (i) generating photo-realistic images given the input geometry and basic intrinsic properties, and (ii) decomposing real images back into their intrinsic components.

Overall, our model follows a similar dual cycle training setup as proposed in [43] and [41]. However, an important conceptual difference to these works is that our task is not a one-to-one but a one-to-many mapping. Different realistic images can be generated from the same set of intrinsic maps as the intrinsics do not uniquely define the image. Likewise, a single image can be explained using different intrinsic decompositions due to projection from the higher dimensional intrinsics into the RGB image space. We therefore introduce a shared adversarial discriminator between the input and the reconstruction at the end of each cycle. We demonstrate that our model enables both highly photo-realistic image synthesis and accurate intrinsic image decomposition. We summarize our main contributions as follows:

- We propose the Intrinsic Autoencoder, a model for learning highly photo-realistic image synthesis with precise control over generated images.
- We propose a method to jointly train image synthesis and intrinsic image decomposition using cycle consistency losses without using any paired data.
- We propose a shared discriminator network that enables better generalization and proves key for learning both tasks without paired training data.
- We systematically analyze the importance of various model components using quantitative metrics and human experiments. We also show that our method recovers accurate intrinsic maps from challenging real images.

We will release our code upon publication.

2 Related Work

Differentiable Rendering. A standard way of synthesizing images from a given geometry and material is to use rendering engines that simulate the physical image formation process. Several works try to implement the rendering process in a differentiable manner, amenable to neural networks. The work of [4] used differentiable rendering with deformable face models for face reconstruction. The works of [31] and [22] proposed rasterization-based differentiable renderers but only support local illumination. In order to support more realistic image formation, some other works [7,12,13,27] propose to back-propagate though path tracing. Differentiable rasterizers are relatively fast, but at the same time highly restrictive as they do not support complex global illumination. While differentiable path tracers produce more realistic images, they are usually quite slow, thus restricting their usage to specific applications. Another drawback of differentiable renderers is that they require a detailed representation of the rendering input in terms of geometry, illumination, materials and viewpoint. In this work, we bypass the specification of complex image formation by training a CNN to directly generate realistic images from given geometry and material inputs.

Neural Image Synthesis. Generative models such as Generative Adversarial Networks [14] and Variational Auto-Encoders (VAE) [24] are widely use to synthesize realistic images from a latent code. In contrast, our goal is to perform conditional image synthesis which allows more fine-grained control over the image generation process. Some popular conditional image generation approaches are label-to-image translation [34,33], image-to-image translation [20,10,8,43,30,38] and text-to-image generation [37,42,16,39]. Earlier works [20,10,8] on conditional image-to-image generation are mostly supervised with paired data from both domains. However, it is challenging to acquire paired training data for some translation tasks such as summer-to-winter image translation. As a result, several works [43,30] propose a way to use unpaired data from both domains for conditional image generation. Other advances in conditional image generation include innovations in network architectures and loss functions for generating high resolution images [38] and generating multiple diverse images [9,18,40,19]. In this work, we develop a new model for photo-realistic geometry-to-image translation using only unpaired training data as supervision. Our work is closely related to [1] which also considers geometry-to-image translation, but requires paired training data. Our work belongs to the family of unpaired conditional image generation models with network architecture and loss functions (e.g., shared discriminator) specialized for the geometry-to-image translation task. We experimentally demonstrate that our model outperforms state-of-the-art unpaired image-to-image translation models [43,18] on this task by a large margin.

Style Transfer. Techniques in this category aim to stylize an input image using the style of a target image while preserving the content of the original input image. Early neural network approaches [11] for style transfer use iterative inference to minimize a style loss between the target image and the output and a content loss between the input image and the output. Later works [17,28,29] proposed feed-forward models for style transfer, thereby speeding up the translation process. The limitation of these techniques is that they translate between two images of similar nature. However, in our case we aim to translate from geometry and material segmentation into an RGB image which are different not just in appearance but also in style.

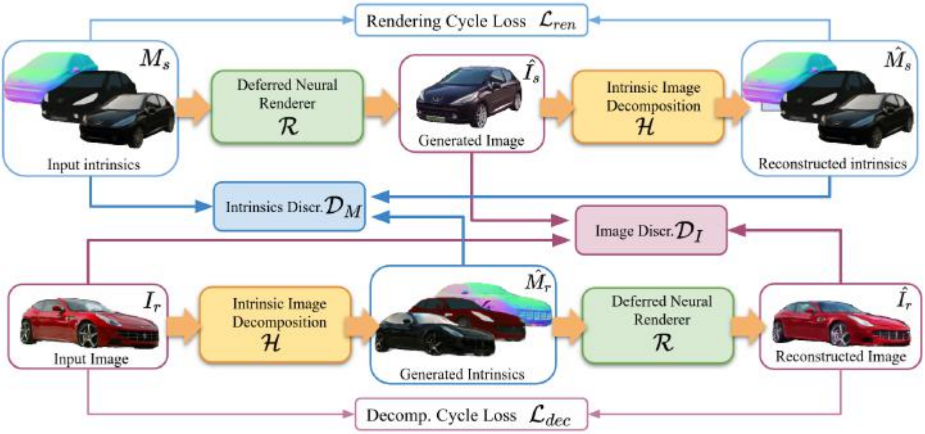

Rendering Cycle Loss $\mathcal{L}_{ren}$
M_s
Input intrinsics
Deferred Neural Renderer $\mathcal{R}$
$\hat{I}_s$
Generated Image
Intrinsic Image Decomposition $\mathcal{H}$
$\hat{M}_s$
Reconstructed intrinsics
Intrinsics Discr. $\mathcal{D}_M$
Image Discr. $\mathcal{D}_I$
I_r
Input Image
Intrinsic Image Decomposition $\mathcal{H}$
$\hat{M}_r$
Generated Intrinsics
Deferred Neural Renderer $\mathcal{R}$
$\hat{I}_r$
Reconstructed Image
Decomp. Cycle Loss $\mathcal{L}_{dec}$

Fig. 2: **Intrinsic Autoencoder.** Our model comprises two cycles: The first cycle (blue) auto-encodes a set of intrinsics rendered from 3D CAD models using appearance as latent representation. The second cycle (red) auto-encodes real images using image intrinsics as representation. Consistency is achieved through a combination of cycle losses and shared adversarial losses. Networks sharing the same weights are illustrated with the same color (green/yellow).

3 Method

Our Intrinsic Autoencoder model (Fig. 2) consists of two generator networks $\mathcal{R}$ and $\mathcal{H}$ for Deferred Neural Rendering and Intrinsic Image Decomposition, respectively. The Deferred Neural Rendering Network $\mathcal{R} : M \to \hat{I}$ takes as input a set of intrinsic maps $M = \{A, N, F\}$. The object's surface normal vectors in the view coordinate system $N \in \mathbb{R}^{H\times W\times 3}$ provides the Deferred Neural Rendering Network important information about the local shape of the object which is necessary for creating shading and reflection in the output image. The albedo $A \in \mathbb{R}^{H\times W\times 3}$ is a pixel-wise RGB value that describes the material or texture color at every pixel, ignoring any lighting effects. Finally, the environment reflections $F \in \mathbb{R}^{H\times W\times 3}$ are computed by projecting a high dynamic range environment map onto the 3D model. Note that this simple projection operation that does not involve any complicated sampling or ray-tracing operations. As shown in our experiments, the Deferred Neural Renderer can also be trained with a subset of those inputs since it is able to compensate for the missing information. The DNR network $\mathcal{R} : M \to \hat{I}$ transforms all the input intrinsics M into a realistic image $\hat{I} \in \mathbb{R}^{H\times W\times 3}$ that corresponds to the input intrinsics. Similarly, the Intrinsic Image Decomposition (IID) network $\mathcal{H} : I \to \hat{M}$ performs the opposite task by taking an input image I and predicting its intrinsics $\hat{M} \in \mathbb{R}^{H\times W\times 9}$.
Supervised training of $\mathcal{R}$ and $\mathcal{H}$ on real data is typically difficult due to the lack of real training image and intrinsics pairs (I_r, M_r). Instead, we use a combination of cycle-consistency losses and adversarial losses that require no paired training examples. This allows us to leverage a large dataset of real images $\{I_r^i\}_{i=0}^n$ and an

unpaired set of synthetically generated intrinsic maps $\{M_s^i\}_{i=0}^m$. In the following, we detail our cycle consistency losses and the novel shared adversarial losses.

3.1 Cycle Consistency

Rendering Cycle. The goal of the rendering cycle is to train $\mathcal{R}$ in order to produce realistic images $\hat{I}_s = \mathcal{R}(M_s)$ from synthetic intrinsic maps M_s. To train $\mathcal{R}$ without paired data, we use the inverse transformation $\mathcal{H}$ which decomposes the predicted image $\hat{I}_s$ back into its intrinsic maps $\hat{M}_s = \mathcal{H}(\mathcal{R}(M_s))$ as illustrated in Fig. 2. We encourage consistency of the intrinsics using the rendering cycle consistency loss which is defined as the *Smooth-*L_1 distance between the input and reconstructed intrinsics

$$\mathcal{L}_{ren}(\mathcal{R}, \mathcal{H}, M_s) = \|\mathcal{H}(\mathcal{R}(M_s)) - M_s\|_1. \tag{1}$$

Decomposition Cycle. Similarly, we train $\mathcal{H}$ to generate intrinsic maps $\hat{M}_r = \mathcal{H}(I_r)$ from real images I_r. To ensure consistency with the input I_r, the output intrinsics $\hat{M}_r$ are passed to the deferred neural renderer $\mathcal{R}$ to reconstruct the image $\hat{I}_r = \mathcal{R}(\mathcal{H}(I_r))$. The decomposition cycle consistency loss is defined by:

$$\mathcal{L}_{dec}(\mathcal{R}, \mathcal{H}, I_r) = \|\mathcal{R}(\mathcal{H}(I_r)) - I_r\|_1. \tag{2}$$

The combined cycle consistency loss is then defined as:

$$\mathcal{L}_{cyc}(\mathcal{R}, \mathcal{H}, I_r, M_s) = \mathcal{L}_{ren}(\mathcal{R}, \mathcal{H}, M_s) + \mathcal{L}_{dec}(\mathcal{R}, \mathcal{H}, I_r) \tag{3}$$

To ensure that the predicted normals $\hat{N}_s = \mathcal{H}_N(I_r)$ and the reconstructed real normals $\hat{N}_r = \mathcal{H}_N(\mathcal{R}(M_s))$ are properly normalized, we exploit an additional normalization loss $\mathcal{L}_{\text{norm}}$:

$$\mathcal{L}_{\text{norm}}(\mathcal{R}, \mathcal{H}, I) = \mid 1 - \|\mathcal{H}_N(I_r)\|_2 \mid + \mid 1 - \|\mathcal{H}_N(\mathcal{R}(M_s))\|_2 \mid$$

3.2 Shared Adversarial Loss

While the cycle consistency loss ensures that the network input can be reconstructed from its output, it doesn't place any importance on the realism of that output. Additionally, the cycle consistency loss assumes a one-to-one deterministic mapping between the input and output. While this is a reasonable condition for some image-to-image translation tasks [43], it is violated when translating between images and their intrinsic properties. Decomposing an RGB image into its high-dimensional intrinsic properties is a one-to-many transformation since multiple decompositions can be consistent at the same time with the same image, e.g., a gray patch may correspond to a gray diffuse surface or a black glossy surface with specular highlight. Likewise, the process of creating an image from an incomplete set of intrinsic properties involves making additional predictions about missing attributes like lighting conditions, optical aberrations, noise or higher-order light interactions. To better capture this multi-modal relationship, we use an adversarial loss between the input and its reconstruction.

An adversarial discriminator $\mathcal{D}$ is a classification model trained to predict if a data sample is produced by a generative model or if it stems from the true data distribution. To train our Intrinsic Autoencoder, we use two adversarial discriminators, $\mathcal{D}_I$ for discriminating generated images $\hat{I}_{\{s,r\}}$ from real images I_r, and $\mathcal{D}_M$ for discriminating generated intrinsic maps $\hat{M}_{\{r,s\}}$ from synthetic intrinsic maps M_s. The discriminators help our model to learn the distribution of real images and synthetic intrinsics by optimizing the following adversarial [14] loss function

$$\mathcal{L}_{\text{adv}}(\mathcal{R}, \mathcal{H}, \mathcal{D}_I, \mathcal{D}_M) = \mathcal{L}^I_{\text{adv}}(\mathcal{R}, \mathcal{H}, \mathcal{D}_I) + \mathcal{L}^M_{\text{adv}}(\mathcal{R}, \mathcal{H}, \mathcal{D}_M) \tag{4}$$

where

$$\mathcal{L}^I_{\text{adv}}(\mathcal{R}, \mathcal{H}, \mathcal{D}_I) = \log(\mathcal{D}_I(I_r)) + \log(1 - \mathcal{D}_I(\mathcal{R}(M_s)) + \log(1 - \mathcal{D}_I(\mathcal{R}(\mathcal{H}(I_r))). \tag{5}$$

is our novel *shared adversarial image loss* which discriminates both between the real image I_r and the generated synthetic image $\hat{I}_s = \mathcal{R}(M_s)$, as well as between the real image I_r and the reconstructed real image $\hat{I}_r = \mathcal{R}(\mathcal{H}(I_r))$. Similarly, we define the *shared adversarial intrinsic loss* as

$$\mathcal{L}^M_{\text{adv}}(\mathcal{R}, \mathcal{H}, \mathcal{D}_M) = \log(\mathcal{D}_{\mathcal{M}}(M_s)) + \log(1 - \mathcal{D}_M(\mathcal{H}(I_r)) + \log(1 - \mathcal{D}_M(\mathcal{H}(\mathcal{R}(M_s)))). \tag{6}$$

Using the reconstructed inputs $\hat{I}_r$ and $\hat{M}_s$ in addition to the generated samples $\hat{I}_s$ and $\hat{M}_r$ for training $\mathcal{D}_I$ and $\mathcal{D}_M$ makes the discriminators more robust and prevents overfitting. This is especially important when a relatively small number of 3D objects are used to create the synthetic intrinsic maps which can lead to a discriminator that recognizes the model features rather that the image realism.

3.3 Implementation and Training

We train our Intrinsic Autoencoder networks $\mathcal{R}, \mathcal{H}$ in addition to the adversarial discriminators $\mathcal{D}_M, \mathcal{D}_I$ from scratch by optimizing the joint objective

$$\min_{\mathcal{R},\mathcal{H}} \max_{\mathcal{D}_I,\mathcal{D}_M} \mathcal{L}_{\text{cyc}} + \mathcal{L}_{\text{norm}} + \mathcal{L}_{\text{adv}} \tag{7}$$

Our framework is implemented in PyTorch [36] and trained using Adam [23] with a learning rate of 0.0002 and a step update schedule. We now describe the architecture of our rendering, decomposition and discriminator networks.

Deferred Neural Rendering Network. We use the coarse-to-fine generator introduced in [38] for the deferred neural rendering network. The input to the network is of size 256 × 512 constructed by concatenating normals, albedo and reflections. The output of the network is an RGB image of size 256 × 512 × 3.

Intrinsic Image Decomposition Networks. We use three networks $\mathcal{H} = \{\mathcal{H}_N,\ \mathcal{H}_A,\ \mathcal{H}_F\}$ for estimating the surface normals N, Albedo A and environment reflections F, respectively, from an image I. For each network, we use a ResNet architecture with 5 ResNet blocks.

Adversarial Discriminator Networks. Since the local structure of the generated images is mostly controlled by the input intrinsics, we want the image discriminator $\mathcal{D}_I$ to mainly focus on the global realism of the output. To address this, we use a multi-scale PatchGAN [38] discriminator which comprises two fully-convolutional networks that classify the local image patches at two scales, full and half resolution. The discriminator outputs a realism score for each patch instead of a single prediction per image. This has been shown to produce more detailed images for similar conditional image generation tasks [20,38,43]. The intrinsics discriminator $\mathcal{D}_M$ has the same architecture except that the input is a 9-channel stack combining all three intrinsic maps. We found that using a single discriminator for the combination of the intrinsic maps performs better than separate networks for each. This is likely due to the inter-dependence between the different intrinsic properties that allows the discriminator to detect inconsistencies between the generated intrinsic maps. We provide more architecture and training details in the supplementary material.

4 Experiments

In this section, we evaluate our networks' performance on deferred neural rendering and intrinsic image decomposition through various qualitative and quantitative experiments.

4.1 Training Data

For training our model, we use two datasets: a synthetic dataset consisting of normals, albedo and reflections of cars, and a real dataset of car images as target.
Synthetic Data Generation. To generate the synthetic training data, we use 3D car models from [2]. The collection contains 28 3D car models covering 6 car categories (SUV, sedan, hatchback, station wagon, mini-van and van). Apart from the geometry, we do not need any high quality or physically-based material or textures for the models. Instead, we assign to each car part a simple material with only two properties, the color and a scalar glossiness factor for computing reflection maps. We ignore the transparency of materials, e.g., glass on the windows, windshield and consider each material to have a single color. We assign each 3D car part a fixed material from a set of 18 fixed materials. Additionally, we randomly sample one of 15 materials with different colors for the car body during the rendering process. Next, a camera position is randomly chosen within a radius of 8 meters and a maximum height of 3 meters. We use a fast OpenGL based rendering engine which operates at around 3 frames per second including the model loading time. It outputs the surface normals of the car model in the camera coordinate space and the albedo channels indicating the material color at each pixel without any lighting or shading. Finally, we produce the environmental reflections by using a 360 degree environment map from [2]. These kind of reflections are very efficient to compute since they only require the view vector and the surface normal and do not rely on expensive path-tracing. Using this setup, we render 20,000 synthetic samples of normals, albedo and reflections.

Real training data. We obtain the real images from a fine grained car classification dataset presented in [25] as our target dataset. For convenience, we refer to this as the real car dataset. It contains 16,000 images of cars captured in various lighting conditions, resolutions and poses and with different camera sensors and lenses. The dataset contains photos of 196 categories of cars from different companies and models. Note that these cars are not paired or registered with the synthetic data. We also observe that only 2 of the car models we use in the synthetic data are present in the real car dataset.

4.2 Baselines

Since our goal is to train with only unpaired data, we choose to benchmark our method against two state-of-the-art unpaired image generation approaches, CycleGAN[43] and MUNIT[18]. However, since both methods were originally designed for image-to-image translation rather than deferred rendering, we setup two additional strong baselines that highlight the importance of our contributions in improving the quality of our results.

CycleGAN and MUNIT. CycleGAN[43] is a generic method for translating between two domains without available paired data. MUNIT[18] aims at producing a diverse set of translations between different domains. We modify the two methods slightly to use our stacked 9 channel synthetic intrinsic maps as inputs. We use the standard code released by the authors to train the methods.

Without shared discriminator. In this setup, we do not use the shared adversarial discriminator discussed in 3.2. Instead, we only use the discriminator $\mathcal{D}_I$ between generated image $\hat{I}_s$, real image I_r. Similarly, the discriminator $\mathcal{D}_M$ is used only between synthetic intrinsics M_s and generated intrinsics $\hat{M}_r$.

Only rendering cycle. Here, we train the model using only the deferred rendering cycle discussed in (Sec. 3.1) and do not use the decomposition cycle.

Ablation of intrinsic maps. In this setup, we train several networks wherein we ablate different input intrinsic maps and train the full model. This is to examine the effect of albedo, normals and reflections on the resulting image quality.

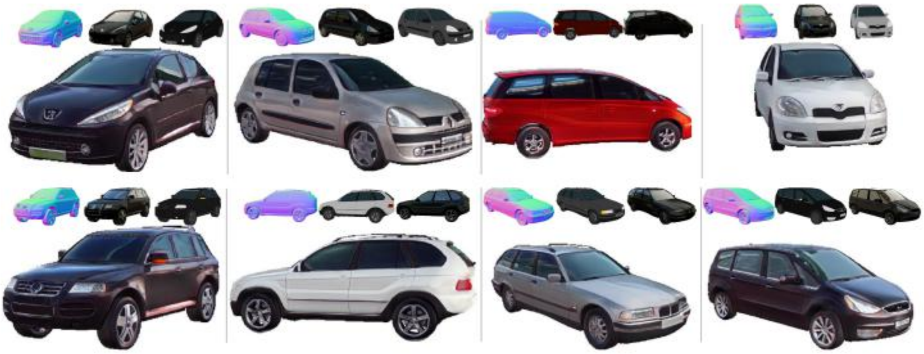


Fig. 3: **Images generated using our Deferred Neural Renderer.** Inputs to the network are intrinsic maps consisting of albedo, normals and reflections, shown above the generated images.

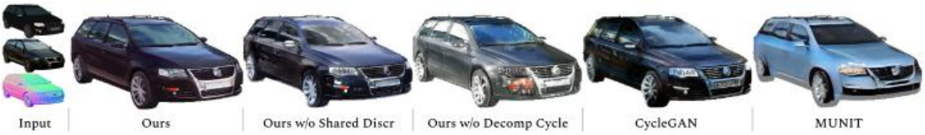


Fig. 4: **Qualitative Comparison with baselines on Neural Rendering.** Inputs to the network are intrinsic maps consisting of albedo, normals and reflections, shown above the generated images. Additional higher resolution results are provided in the supplementary materials.

4.3 Deferred Neural Rendering

In this section, we evaluate our approach for the task of deferred neural rendering. For this experiment, we use the network $\mathcal{R}$ to produce images given synthetic intrinsic maps (albedo, normals, reflections). We also compare our results to other baselines, both qualitatively and quantitatively.

4.3.1 Qualitative results

Fig. 3 shows car images generated using our deferred neural renderer from the input synthetic intrinsic maps shown above them. The car models in the evaluation set have been previously seen by the generator, but the unique combination of pose and paint color has not been seen during training. Our approach is able to generate detailed photo-realistic images of cars with consistent geometry and distinct parts. We emphasize that the deferred neural rendering network is trained without any rendered or real geometry-image pairs. Instead, it is able to learn the appearance of different car parts from a large set of real car images.
In Fig. 4 we compare the results of our full model to various baselines. The results clearly show the improvements in visual quality achieved when using our full model. Specifically, MUNIT appears to be unable to preserve the geometry and albedo of the input in the generated image, CycleGAN image has significant artefacts on the windows, body, etc., hence suffering from poor image quality. When we train our model without the shared discriminator, the resulting images suffer from irregular reflection patterns and a noisy image. This is likely due to the strong overfitting required by the network to reproduce the input image exactly when using only an L_1 loss. The model trained without the decomposition cycle is not able to preserve the input intrinsics in terms of albedo and reflection. Adding the decomposition cycle and shared discriminator alleviates these problems to help the network produce a high quality photorealistic image while also maintaining fidelity to input intrinsics.

In figure 5, we show the effect of input intrinsic maps on the quality of rendered images. When the model is trained only with normals as intrinsic input, the geometry of the result is well rendered but the color of different parts poorly defined. The model trained on both normals and albedo demonstrates sharper image quality but the hallucinated reflections by the network lacks lack realistic details. Finally, using the environmental reflections helps the network produce consistent and realistic images with sharp details.

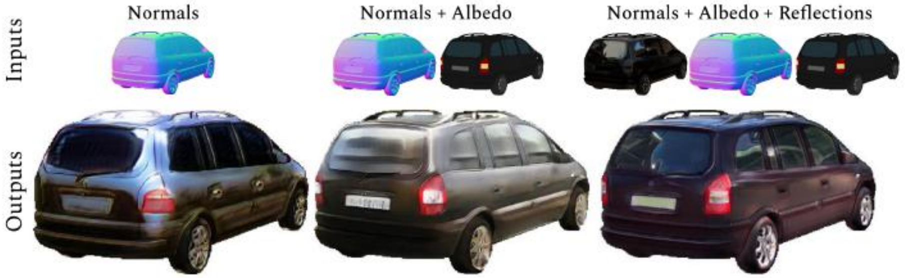


Fig. 5: **Images generated using models trained with ablated inputs.**

4.3.2 Quantitative results

We evaluate the quality of generated images using Fréchet Inception Distance(FID) [15] and Kernel Inception Distance(KID) [3]. Both metrics compute the distance between the features of a set of real images and generated images, obtained from a pretrained CNN. To compute these metrics, we first generate a set of 12,000 intrinsic maps as described in section 4.1 to be used as input data. We use the same input data for each of the baselines to verify the resulting image quality. For real image samples, we use the real training data mentioned in 4.1.

Table 1 presents both the FID and KID between the images generated using various methods and the real samples. Our full model has the lowest distances (47.6, 4.2) indicating that the rendered images from our model are closest to the distribution of real images. Both the existing state-of-the-art conditional image generation models, CycleGAN and MUNIT have higher distances suggesting that the quality of their generated images is lower than ours validating what has been observed in qualitative results. Further, when we ablate each of the intrinsic map inputs, both FID and KID increase substantially. Notably, in the case of ablating albedo input, the highest increase in distances can be observed (88.7, 5.4), implying its importance for photo-realistic image generation. Similarly, ablating normals or reflections also increases the distances significantly. We conclude that albedo is the most important for our task followed by normals and reflections maps. In both cases where we ablate the decomposition cycle or rendering cycle, we observe a huge increase in the distances signifying the importance of using both cycle consistency losses during training. Finally, training with the setup of separate discriminators as mentioned in 4.2 leads to an increase in the distances.

Setup	Cycle GAN	MUNIT	**Our Model**	w/o Shared Discr.	w/o Decom. Cyc.	w/o *A*	w/o *N*	w/o *F*
FID	103.3	99.0	**47.6**	59.2	99.6	88.7	60.2	56.7
KID	10.2	13.5	**4.2**	4.8	11.8	5.4	4.9	5.9

Table 1: **FID and KID between real images and generated samples.** All inputs are provided to the generator (Albedo, Normals and Reflections).

4.3.3 Human Experiments

In this section we design two experiments to measure the visual realism of generated car images. We leverage Amazon Mechanical Turk to crowd source human evaluations. For each comparison, we presented 40 human subjects each with 50 image pairs to choose the more realistic looking image. The results are presented in Table 2. The first row presents experiments where one image is picked from the real images and the other is from one of the synthesis methods and presented in a random order. Images from our full model seem to be most confused with real car image since only 67.5% of choices were correct while in 32.5% of the trials the subjects choose our images to be the real one.
In the second experiment subjects are presented with an image generated by our full model and a matching image generated by one other synthesis methods. The results in the second row of Table 2 show that subjects choose our results to be more realistic over 80% of times when compare to CycleGAN and MUNIT. This clearly indicates a high level of visual quality of our generated images compared to those generated from existing methods. On the other hand, images from our ablated models appear to be much closer to our full model visual quality.

Setup	Cycle GAN	MUNIT	**Our Model**	w/o Shared Discr.	w/o Decom. Cyc.
Real Images	77.7%	75.6%	**67.5%**	68.9%	71.0%
Our Model	80.0%	85.8%	–	57.6%	63.8%

Table 2: **Human Subject Study.** Pairwise comparisons to identify realistic images in an A/B test using Amazon Mechanical Turk. The numbers indicate the ratio of trials where the image produced by the method in side was chosen as more realistic compared to the image from the method on the header.

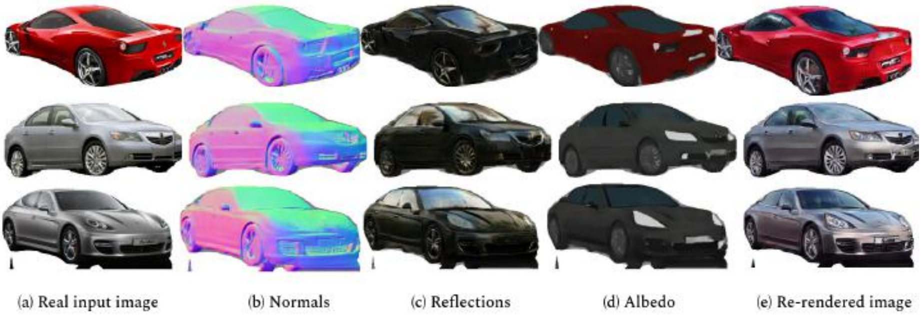
4.4 Intrinsic Image Decomposition

In this section, we evaluate our approach for the task of intrinsic image decomposition. We use the network $\mathcal{H}$ to decompose a given image into its intrinsic maps (albedo, normals, reflections). We also compare our results to other baselines.

4.4.1 Qualitative results

In fig. 6, we show that the intrinsic decomposition network is able to decompose real car images into their intrinsic maps. We would like to emphasize that the model does not have access to ground-truth intrinsic maps for real images during the training phase. Also, these car models are not present in the synthetic training data. Although there are no race car models in the training data, the model generalizes to new geometries of cars. The last column in fig. 6 shows the re-rendered images produced by the deferred rendering network using the predicted intrinsic maps as input.

Figure 7 compares the decompositions produced by our model to those from other baselines. Both CycleGAN and MUNIT do not generalize well to real images. MUNIT is unable to recover any of the intrinsic maps. The reflection maps



(a) Real input image (b) Normals (c) Reflections (d) Albedo (e) Re-rendered image

Fig. 6: **Results of our intrinsic decomposition network on real images.** The first column shows the inputs to the network. Our model is able to decompose the sport car in first row accurately even though our synthetic training dataset doesn't include any sport cars at all. The car models of other inputs images are also not present in our synthetic dataset.

predicted by CycleGAN contain heavy artefacts. Our model without decomposition cycle also recovers noisy albedo and normals. This is because training the model without the decomposition cycle leads the networks to overfit only to synthetic data leading to poor generalization on real data. On the other hand, training without the shared discriminator leads to severe artefacts in the decompositions. This is because the rendering network tries to encode intrinsics information in the generated images in the form of high frequency artefacts such that the decomposition network can easily recover them. This leads to poor generalization of the decomposition network on real data.

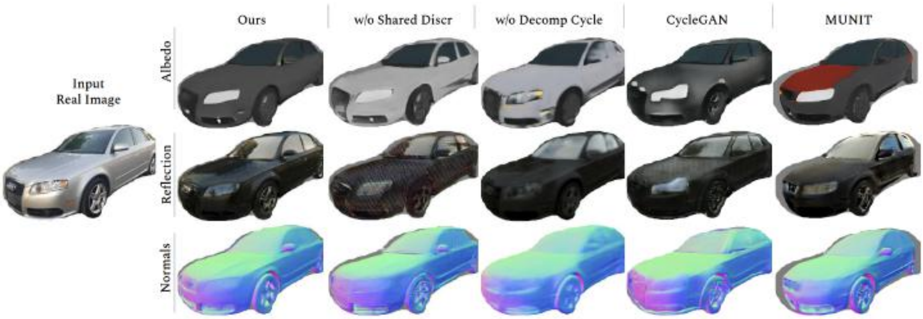


Fig. 7: **Comparison with Baselines for intrinsic decomposition.**

4.4.2 Quantitative results

We evaluate the intrinsic maps (normals, albedo, reflections) predicted by the intrinsic image decomposition network ($\mathcal{H}$). For this we construct a synthetic dataset containing RGB images and its corresponding intrinsic maps. We obtain the synthetic images using a standard Physically Based Renderer(PBR) (Blender [5]) using the synthetic intrinsic maps. We use such a setup because there is no existing dataset which contains real images annotated with the required intrinsic maps. To obtain the error between predicted and ground truth normals, we compute the average cosine distance between them. The errors for albedo and reflection are the average ℓ_1 distances between the predicted and ground truth maps. Table 3 presents the errors of various methods for predicting intrinsic maps. Our full model has the least error for all the modalities followed by our model without the shared discriminator and without decomposition cycle. Note that these PBR-rendered images have not been presented to our network during training. On the other hand, the errors for CycleGAN and MUNIT are significantly higher. This indicates that our model is able to learn accurate image decomposition while generalizing across various kinds of images (real, PBR-rendered, network generated) even in the absence of paired ground truth data.

	w/o Shared Discr.	w/o Decom. cycle	CycleGAN	MUNIT	Full Model
Normal Error	17.75°	18.80°	27.82°	29.15°	**14.73°**
Albedo Error	54.00	67.21	68.18	81.44	**52.74**
Reflection Error	55.60	71.00	73.18	74.75	**51.74**

Table 3: **Results on the Intrinsic Decomposition Task.** Our method achieves the lowest decomposition error on all 3 intrinsic maps.

4.5 Results on ShapeNet Aeroplanes

In this experiment, we train our model for the object class "Aeroplanes". For this experiment, we obtain the real images from FGVC-Aircraft dataset introduced in [32] which contains 10,000 images of aeroplanes. We use the 3D models of aeroplanes from the Shapenet dataset [6] to obtain our intrinsic maps. We follow the process mentioned in sec.4.1 to generate input training data. We use the normals and albedo as inputs to the network. Figure 8 illustrates realistic images generated using our deferred rendering network, demonstrating the ability of our method to handle low-quality mesh and texture models.

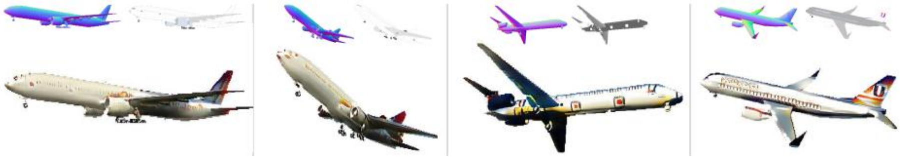


Fig. 8: **Images generated by our network trained on aeroplanes.**

5 Conclusion

In this paper, we presented a joint approach for training a deferred rendering network for generating realistic images from synthetic image intrinsics and an intrinsic image decomposition network for decomposing real images of an object into its intrinsic properties. We trained the model using unpaired 3D models and real images. Our qualitative and quantitative experiments revealed that using a combination of shared adversarial losses and cycle consistency losses is able to produce images that are both realistic and consistent with the control input.

References

1. Hassan Abu Alhaija, Siva Karthik Mustikovela, Andreas Geiger, and Carsten Rother. Geometric image synthesis. *Asian Conference on Computer Vision (ACCV)*, 2018.
2. Hassan Abu Alhaija, Siva Karthik Mustikovela, Lars Mescheder, Andreas Geiger, and Carsten Rother. Augmented reality meets computer vision: Efficient data generation for urban driving scenes. *International Journal of Computer Vision (IJCV)*, 2018.
3. Mikoaj Bikowski, Dougal J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. In *International Conference on Learning Representations*, 2018.
4. Volker Blanz, Thomas Vetter, et al. A morphable model for the synthesis of 3d faces. In *Siggraph*, volume 99, pages 187–194, 1999.
5. Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Blender Institute, Amsterdam,
6. Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher 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.
7. Chengqian Che, Fujun Luan, Shuang Zhao, Kavita Bala, and Ioannis Gkioulekas. Inverse transport networks. *arXiv preprint arXiv:1809.10820*, 2018.
8. Qifeng Chen and Vladlen Koltun. Photographic image synthesis with cascaded refinement networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1511–1520, 2017.
9. Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8789–8797, 2018.
10. Alexey Dosovitskiy, Jost Tobias Springenberg, Maxim Tatarchenko, and Thomas Brox. Learning to generate chairs, tables and cars with convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(4):692–705, 2017.
11. Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2414–2423, 2016.
12. Ioannis Gkioulekas, Anat Levin, and Todd Zickler. An evaluation of computational imaging techniques for heterogeneous inverse scattering. In *European Conference on Computer Vision*, pages 685–701. Springer, 2016.
13. Ioannis Gkioulekas, Shuang Zhao, Kavita Bala, Todd Zickler, and Anat Levin. Inverse volume rendering with material dictionaries. *ACM Transactions on Graphics (TOG)*, 32(6):162, 2013.

14. Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.
15. Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6626–6637. Curran Associates, Inc., 2017.
16. Seunghoon Hong, Dingdong Yang, Jongwook Choi, and Honglak Lee. Inferring semantic layout for hierarchical text-to-image synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7986–7994, 2018.
17. Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510, 2017.
18. Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 172–189, 2018.
19. Le Hui, Xiang Li, Jiaxin Chen, Hongliang He, and Jian Yang. Unsupervised multi-domain image translation with domain-specific encoders/decoders. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 2044–2049. IEEE, 2018.
20. Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.
21. Wenzel Jakob. Mitsuba renderer, 2010. http://www.mitsuba-renderer.org.
22. Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3907–3916, 2018.
23. Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.
24. Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
25. Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.
26. Christoph Lassner, Gerard Pons-Moll, and Peter V. Gehler. A generative model for people in clothing. In *ICCV*, 2017.
27. Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. Differentiable monte carlo ray tracing through edge sampling. In *SIGGRAPH Asia 2018 Technical Papers*, page 222. ACM, 2018.
28. Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Universal style transfer via feature transforms. In *Advances in neural information processing systems*, pages 386–396, 2017.
29. Yijun Li, Ming-Yu Liu, Xueting Li, Ming-Hsuan Yang, and Jan Kautz. A closed-form solution to photorealistic image stylization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 453–468, 2018.
30. Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *Advances in Neural Information Processing Systems*, pages 700–708, 2017.
31. Matthew M Loper and Michael J Black. Opendr: An approximate differentiable renderer. In *European Conference on Computer Vision*, pages 154–169. Springer, 2014.

32. S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. Technical report, 2013.
33. Takeru Miyato and Masanori Koyama. cgans with projection discriminator. *arXiv preprint arXiv:1802.05637*, 2018.
34. Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2642–2651. JMLR. org, 2017.
35. Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 2337–2346. Computer Vision Foundation / IEEE, 2019.
36. Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.
37. Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. *arXiv preprint arXiv:1605.05396*, 2016.
38. Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8798–8807, 2018.
39. Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1316–1324, 2018.
40. Zichen Yang, Haifeng Liu, and Deng Cai. On the diversity of realistic image synthesis. *arXiv preprint arXiv:1712.07329*, 2017.
41. Zili Yi, Hao Zhang, Ping Tan, and Minglun Gong. Dualgan: Unsupervised dual learning for image-to-image translation. *CoRR*, abs/1704.02510, 2017.
42. Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5907–5915, 2017.
43. Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2223–2232, 2017.