No Shadow Left Behind: Removing Objects and their Shadows using Approximate Lighting and Geometry

Edward Zhang[1] Ricardo Martin-Brualla[2] Janne Kontkanen[2]
Brian Curless[1,2]
[1]University of Washington [2]Google

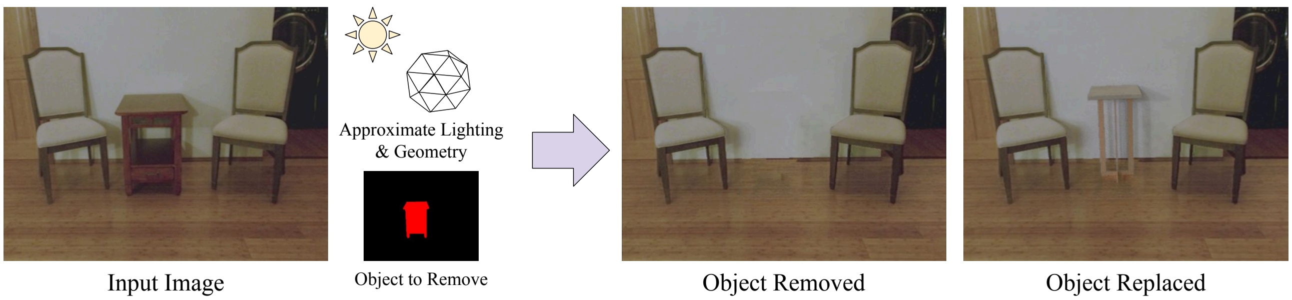


Figure 1: We present a method to remove an object and its shadows from an image, to enable applications like home refurnishing. Our method takes as input an image, approximate scene lighting and geometry, and an object mask, and generates a new version of the image that depicts the scene as if the object had not been present. This not only includes inpainting the occluded pixels, but removing any shadows cast by the object.

Abstract

Removing objects from images is a challenging technical problem that is important for many applications, including mixed reality. For believable results, the shadows that the object casts should also be removed. Current inpainting-based methods only remove the object itself, leaving shadows behind, or at best require specifying shadow regions to inpaint. We introduce a deep learning pipeline for removing a shadow along with its caster. We leverage rough scene models in order to remove a wide variety of shadows (hard or soft, dark or subtle, large or thin) from surfaces with a wide variety of textures. We train our pipeline on synthetically rendered data, and show qualitative and quantitative results on both synthetic and real scenes.

1. Introduction

Mixed reality aims to seamlessly combine the virtual and the real. As one example, imagine an interior design app that lets you try out new furniture. Most previous work in augmented reality focuses on inserting virtual objects – for instance, putting a virtual sofa into your living room. The scope of these applications can be greatly expanded by also enabling manipulation of real-world objects – imagine removing the futon that you intend to replace with the sofa, and moving a coffee table over to make more room for it.

Previous work on object removal has focused solely on the inpainting problem – that is, replacing the pixels previously occupied by the removed object. However, for realistic results, we need to remove the sofa *and* the shadows it casts on the wall and the floor, as well as the reflection on the hardwood floor. For the purposes of this paper, we focus only on the shadow removal problem.

Existing inpainting-based approaches for object removal either ignore the shadows of the object, or mark them to be inpainted as well. However, very large shadows may leave little image content to copy pixels from. Furthermore, this approach requires segmenting out the object's shadow in addition to the object itself – a difficult task, as varying lighting conditions can cause multiple shadows, very soft shadows, or overlapping shadows, and a surface texture may have dark regions that could be mistaken for shadows.

Inspired by Debevec's [9] work in virtual insertion of objects in scenes, we use a scene proxy to help determine the visual effects of a scene manipulation. Debevec performs the scene edit on the proxy model, and renders the proxy pre- and post-edit. The pixelwise difference between the two renderings, which for object insertion contains shadows and reflections of the virtual objects, is then applied to the input image to produce the final output. This method is known as *differential rendering*. However, it is not practical to solve the shadow removal problem by applying the pixelwise difference directly, since the shadows in the proxy model are only a rough estimate of the real shadows. To account for this, we propose a neural network based system for more general differential rendering for object removal.

An obvious question is, how do we obtain an editable scene proxy? One could use a depth camera, monocular depth estimation [13, 14], or a global model obtained as a side effect of localization [8] for the geometry. For lighting, the possibilities include a mirror sphere, panorama, or learning based methods [22, 29, 30]. In this paper we use depth maps captured by an affordable depth sensor and a 360° panorama, but the method is not fundamentally limited to proxies obtained by these devices, nor do the proxy models need to be very accurate. Our proxy mesh is generated from a single depth map, thus modeling only front facing surfaces, and our lighting is captured as an uncalibrated HDR environment map with only very rough alignment. We show that even this constrained and incomplete proxy provides enough information to generate plausible removal results across a wide range of conditions.

In this paper we present a method for removing an object and its shadows from an input image, given a rough model of the scene and the mask of the object. Our system is more accurate and produces fewer visual artifacts than a general image-to-image translation system or an inpainting method, even when the inpainting method is given the shadow regions it should replace.

2. Related Work

2.1. Scene Editing

Editing scenes in a visually realistic manner has long been an area of interest in the graphics community. Most of this work has focused on virtual object insertion. Classical methods construct an approximate model of the scene to help perform these edits, ranging from Debevec's early work [9], which assumes lighting and geometry were directly captured, to more recent work by Karsch et al. [19], which infers geometry, albedo, and lighting from a single image. Beyond simply inserting objects, Kholgade *et al.* [20] are able to move an object around, although they assume that a high-quality 3D model of the object is available.

Research on object removal has traditionally focused on the inpainting problem, ranging from classical techniques such as PatchMatch [1] to recent learning-based techniques such as DeepFill [36] and HiFill [35]. These methods do not consider lighting interactions between the removed object and the rest of the scene; thus when removing the object by inpainting, the user-specified mask must be extended to include the object's shadow. Recent work by Wang *et al.* [34] employs deep networks to associate shadows with their casters; however, their instance segmentation approach produces hard boundaries and does not work for soft shadows. Zhang *et al.* [37] remove objects from indoor scenes by constructing a full scene model and rendering it without the objects, eliminating the need for inpainting and shadow identification; however, their approach requires an involved capture process and is limited by the expressivity of their parametric scene model.

The issue of the limited range of scene models is inherent to all of the methods that rely on such models for scene editing. Most works (e.g. Kholgade [20]) use Debevec's differential rendering method to account for differences between the model and the real scene. Recent approaches for neural rerendering use image-to-image translation to map from the domain of the approximate model to a realistic result [23, 24]. Philip *et al.* [26] introduced a method for relighting outdoor photographs that also leverages proxy scene models. Their system is designed to handle global changes in illumination, like changing the position of the sun; furthermore they rely heavily on shadow masks that cannot handle complex environment lighting. Instead, we focus on local changes informed by the differences in the appearance of the proxy scene, based on an intrinsic decomposition that can handle multiple soft shadows.

2.2. Shadow Removal

Removing shadows from images is another problem that has a long history. Note that the goal of these works is to remove all shadows from an image, while our goal is to isolate and remove the shadow(s) of a single object. To help approach this more challenging task, we assume the presence of a rough scene proxy.

Classical intrinsic image decomposition methods are designed with various priors, typically specializing in low-frequency lighting and thus handling soft shadows well [3, 2, 5, 15]. Another set of methods specialize in hard shadows and classify gradients as shading or texture [4, 31]; however, these methods break down when shadow receivers have complex texture. Finlayson *et al.* [11] place assumptions on light source chromaticity, allowing for removal of both soft and hard shadows at the expense of generality.

Recent methods use deep networks to perform shadow detection and removal, starting with work by Qu *et al.* [27]. Advances such as adversarial losses [33, 10], a two-stage detection-then-removal scheme [16], or lighting inference [21] have resulted in great improvements on shadow removal on the common ISTD [33] and SRD [27] datasets. However, these datasets only contain hard shadows produced from outdoor lighting. Our system is trained to handle much more diversity in lighting conditions.

3. Training Data

To better understand our architecture and losses, we first discuss our training data. Our system is trained on a synthetic dataset, which allows us to greatly expand the diversity of lighting and receiver textures compared to prior datasets. Furthermore, it also allows us to generate ground truth intermediates such as intrinsic decompositions, which are crucial to our system.

To generate training data, we set up 60000 input scenes with randomly generated geometry, textures, lighting, and camera parameters. These scenes are rendered using PBRT [25] to produce images of resolution 512×512. The dataset exhibits a wide variety of shadow casters (e.g large objects, thin structures, and objects with unusual silhouettes) and lighting conditions (hard or soft shadows, very dark or very subtle shadows, multiple shadows). Some examples are shown in Figures 3-4 and in the supplementary material.

3.1. Scene Generation

Geometry: Our generated scenes consist of a ground plane supporting six to seven objects randomly selected from the ~50000 3D models in the ShapeNet [6] dataset, which include a variety of object classes ranging from furniture and tableware to cars and airplanes. These objects are arranged in a ring around a central object, and are scaled such that the bounding boxes are nonintersecting. Each object is translated such that it lies entirely on top of the plane, and has a random rotation around its up axis. The ground plane is large enough to support all the shadow casters, plus an additional margin for shadows to potentially fall upon.

Materials: The supporting plane is given a matte material, and is assigned a random texture (e.g. carpet, wood, stone, tile). Existing texture datasets are too small (e.g. Brodatz [32]) or have textures which are nonuniform (e.g. Describable Textures Dataset [7]). We use a manually curated texture dataset of ~8000 images from Google Image Search results. ShapeNet objects come with prespecified materials.

Lighting: We illuminate each scene by one of the ~400 HDRI maps at HDRI Haven[1], randomly rotated around the up axis. To supplement the lighting, we add a point light with random intensity (setting the maximum to the peak intensity of the HDRI map) randomly placed between a minimum and maximum distance from the center of the plane, in the upper hemisphere.

Camera: We define a range of camera positions lying on an upper hemisphere of fixed radius in terms of spherical coordinates facing the center of the scene. We allow the azimuthal angle to vary freely, but set a minimum and maximum elevation angle (as people rarely observe scenes from directly overhead or from very low angles). After selecting an initial camera pose we then perturb the camera's position while keeping the same orientation.

[1] https://hdrihaven.com/

3.2. Image Generation

Using PBRT, we render three RGB images of each scene: $\hat{T}$, the ground plane alone with diffuse texture; $\hat{L}$, the complete scene with the plane material replaced by a diffuse white material; and $\hat{L}'$, the scene with the central object removed with the same alteration to the plane material. These images comprise the ground truth intrinsic decomposition's texture and lighting, and the lighting post-object-removal. Note that these images do not form a true intrinsic decomposition of the entire scene, only of the receiving plane. We also render the ground plane alone with no texture to capture its unshadowed appearance L_r.

Next, we render depth maps D, D' of the unedited and target scenes, as well as a depth map D_r of solely the ground plane receiving shadows, all using the same camera pose as the RGB images. From these we compute a pixel mask of the object to be removed $M_o = \mathbb{I}(D' < D)$ which is 1 where the object is and 0 everywhere else. We also compute a receiver mask $M_r = \mathbb{I}(D_r \neq \infty, D_r = D)$ which is 1 for pixels lying on the ground plane in the unedited scene and 0 everywhere else.

To allow for further augmentation, we do not raytrace the input and output images $I, \hat{I}'$; instead we compute them at train time from the decomposition: $I = (M_r\hat{T}+(1-M_r))\hat{L}$ and $\hat{I}' = (M_r'\hat{T} + (1 - M_r'))\hat{L}'$. This allows us to modify the hue, saturation, and brightness of texture and lighting at train time. Note that we forgo indirect bounce lighting in our synthetic data to enable this augmentation, as indirect illumination depends on the surface reflectance, *i.e.* texture.

To mimic real capture, we add noise to the depth map of the unedited scene and construct a triangle mesh from the depth map as our approximate geometry, replacing the ground plane vertices with a best-fit plane (which continues behind the removed object). To form the target proxy geometry, we delete the depth pixels occupied by the removed object. This scene is lit with a perturbed version of the input lighting: we jitter the point light's position, color, and intensity, apply a random nonlinear scale to the HDRI map, and randomly rotate the HDRI map by a small amount. All materials are set to a diffuse white; note that we do not model the surface reflectance (texture) of the plane as it would imply already knowing the intrinsic decomposition. Rendering these elements produces P, P', respectively the images of the unedited and target scene proxy.

3.3. Normalization

Our intrinsic decomposition has a scale ambiguity, which we resolve by normalizing the ground truth lighting $\hat{L}, \hat{L}'$ at train time, expecting that the network will produce normalized lighting images. Specifically, we apply a per-channel scale to both $\hat{L}, \hat{L}'$ such that the maximum pixel

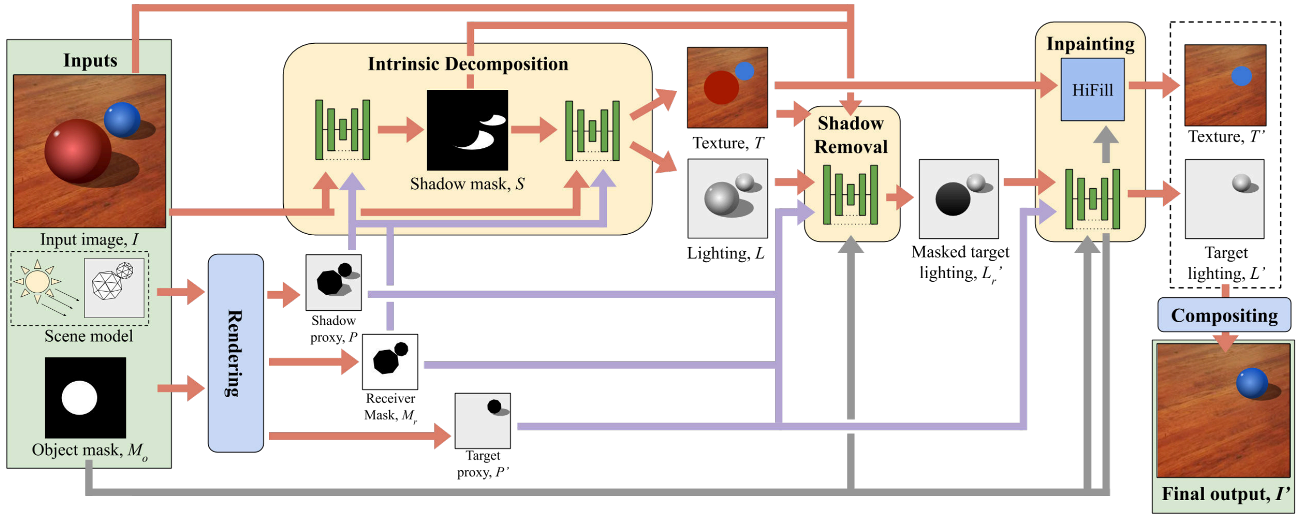


Figure 2: Overview of our approach. We take an input image, a mask of the object to remove, and renderings of a rough scene model with and without the object. We first decompose the input image into texture and lighting. We then remove the object's shadows from the lighting image. We inpaint the object mask region separately in both texture and lighting before recompositing to get our final result.

value on the receiver across both images is $(1, 1, 1)$. Similarly, we compute a normalization factor for images of the scene proxy P, P' (which are just approximations of $\hat{L}, \hat{L}'$). This occurs at both test and train time.

For both train and test we scale the images in the input domain (i.e. I and $\hat{I}'$) to have a channelwise mean pixel value of 0.5 on the ground plane.

4. Method

Our pipeline consists of a series of convolutional neural networks (we use a single U-Net [28] for each component), with an inpainting stage to produce our final results. A visual overview can be seen in Figure 2. Our system takes as inputs the original image, a rendering of the approximate scene model before object removal (referred to as the *shadow proxy*), a rendering of the scene model after object removal (*target proxy*), and binary masks denoting the object to remove and the receiving surface from which to remove shadows. An overview of our pipeline follows:

- The **intrinsic decomposition subsystem** separates the input image into texture and lighting, guided by the shadow proxy. Following existing works [16] on shadow removal, we use a two-stage scheme with an initial shadow segmentation network.
- The **shadow removal** network removes the shadow of the removed object from the decomposed lighting, aided by the shadow proxy and target proxy images.
- The **inpainting subsystem** separately inpaints the lighting and texture behind the removed object. Our learned lighting inpainting uses the target proxy to inform where the remaining shadows in the scene should continue behind the removed object. For texture inpainting, we use an off-the-shelf inpainting method. Inpainting the decomposed texture, rather than the final composite, prevents the inpainting method from hallucinating its own shadows.
- Lastly, we recompose the lighting and texture images back together to produce our final result.

We define I, I' as the input and output images, respectively. P, P' are the shadow proxy and target proxy. The intrinsic decomposition is denoted by $I = LT, I' = L'T'$ with L, L' being the lighting images and T, T' being the reflectance (texture) images. M_o is a binary mask which is 1 for pixels lying on the object to be removed and 0 elsewhere. M_r is a binary receiver mask which is 1 for pixels lying on the local scene receiving the shadows, and 0 for pixels elsewhere. This restricts the network to operate on a surface with a single texture, as otherwise the intrinsic decomposition frequently mislabels shadows as changes in reflectance if the surface reflectance can vary arbitrarily.

RGB images are processed in the log domain, turning the intrinsic decomposition $I = LT$ into a sum $\log(I) = \log(L) + \log(T)$ that is more naturally represented by CNNs. Using synthetic training data enables full supervision of each subnetwork's intermediate outputs.

Shadow Segmentation: This subnetwork produces a 1-channel soft segmentation in $[0, 1]$, where 1 is full shadow.

$$S = f_{\text{SS}}(\log(I), \log(P), M_r) \tag{1}$$

Intrinsic Decomposition: This subnetwork decomposes the input image into 3-channel lighting L and texture T.

$$L, T = \exp(f_{\text{ID}}(\log(I), \log(P), S, M_r)) \tag{2}$$

Shadow Removal: This subnetwork removes the shadow of the object from the predicted lighting image, producing one 3-channel output, the masked target lighting.

$$\begin{aligned} L'_r = \exp(f_{\text{SR}}(&\log(I), \log(T), \log(L), \\ &\log(P), \log(P'), S, M_r, M_o)) \end{aligned} \tag{3}$$

Lighting Inpainting: This subnetwork fills in the predicted lighting L'_r behind the removed object, continuing shadows cast by other objects through the mask if necessary. The target lighting L' is then the composite of the inpainted lighting and the masked target lighting.

$$L'_o = \exp(f_{\text{LI}}(\log(L'_r), \log(P'), M_o)) \tag{4}$$

$$L' = (1 - M_o)L'_r + M_o L'_o \tag{5}$$

Texture Inpainting: We inpaint the texture image using an inpainting operator $g(T, M_o)$, synthesizing the pixels of T in the hole region specified by the mask image M_o. For our experiments we used HiFill [35] for $g(T, M_o)$, trained on the Places2 dataset [40].

$$T' = g(T, M_o) \tag{6}$$

Final Composite: The previous stages predicted the appearance of the receiver within the target receiver mask $M'_r = M_r + M_o$. We composite the remaining pixels from the original image, consisting of unaffected surfaces beyond the local scene and other objects within the local scene.

$$I' = M'_r T' L' + (1 - M'_r)I \tag{7}$$

4.1. Training

Each subnetwork is independently trained with the Adam optimizer for 60 epochs on 60000 training scenes, with ground truth intermediates substituted for the outputs of earlier subnetworks. We then train the whole system end-to-end for 60 epochs. The system was implemented in Tensorflow and trained on four Tesla V100 GPUs with a batch size of 16. The learning rate was 10^{-4} decaying by 0.5 every 10 epochs.

Our loss functions are described below. For all networks except the lighting inpainting network, the inputs to the losses are masked with M_r to only apply to pixels lying on the receiver. For brevity we assume that the norm flattens across input channels and image dimensions. We denote a ground truth supervision image with a hat, so that the ground truth intrinsic decomposition is $\hat{L}, \hat{T}$, ground truth output image is $\hat{I}'$, and so on.

Shadow Segmentation: It is difficult to define a ground truth for what constitutes a shadow in a scene lit by an HDRI map, since any object will occlude some part of the distant illumination. To supervise this stage we therefore examine the ratio of pixel values between the ground truth lighting with all the objects compared to the ground truth lighting of only the receiving surface. A shadow is defined as any pixel where this ratio is is less than the median ratio on any of the three color channels using a soft threshold:

$$\hat{S} = \max\left(\sigma\left(\frac{\text{median}(\hat{L}/L_r) - \hat{L}/L_r}{\alpha}\right)\right) \tag{8}$$

The shadow segmentation subnetwork is supervised by a class-balanced binary cross entropy term as well as a loss on the gradients of the shadow segmentation:

$$E_{\text{SS}} = \lambda_S E_S + \lambda_{\nabla S} E_{\nabla S} \tag{9}$$

$$E_S = \frac{-\hat{S}\log(S)}{||\hat{S}||_1} - \frac{(1-\hat{S})\log(1-S)}{||1-\hat{S}||_1} \tag{10}$$

$$E_{\nabla S} = ||\nabla S - \nabla \hat{S}||_2 \tag{11}$$

Intrinsic Decomposition: The intrinsic decomposition loss function is the most involved of our losses. The effects of each term, as well as comparisons to existing works in intrinsic decomposition and shadow removal, are shown in the supplementary material.

$$E_{\text{ID}} = \lambda_{LT} E_{LT} + \lambda_{\text{excl}} E_{\text{excl}} + \lambda_I E_I + \lambda_{\nabla L} E_{\nabla L} \tag{12}$$

For the data term, a multiscale loss on the predicted lighting and texture images was vital to ensure the model would work well on high-contrast textures.

$$E_{LT} = P(L, \hat{L}) + P(T, \hat{T}) \tag{13}$$

where $P(X, \hat{X})$ is an L2 loss on a Gaussian pyramid decomposition of the images $X, \hat{X}$.

To ensure a clean decomposition, we impose the exclusion losses of Zhang *et al*. [39] on the predicted lighting and texture images, which in essence constructs 0-to-1-valued edge maps at multiple scales, and penalizes edges lying at the same location in the two decomposed images.

$$E_{\text{excl}} = \sum_{i=0}^{i=3} 4^i ||\Psi(T \downarrow i, L \downarrow i)|| \tag{14}$$

where $X \downarrow n$ denotes image X downsampled bilinearly by a factor of 2^n, and Ψ is as defined by Zhang et al.

We also have an L1 loss on the two decomposed images recomposing into the input image.

$$E_I = ||I - LT||_1 \tag{15}$$

	Synthetic			Real		
	RMSE	Shadow RMSE	Inpaint RMSE	RMSE	Shadow RMSE	Inpaint RMSE
No-op	0.0455	0.2785	0.3755	0.0405	0.1413	0.2702
PatchMatch	0.0460	0.2790	0.2565	0.0401	0.1378	0.1288
HiFill	0.0479	0.2700	0.2555	0.0408	0.1305	0.1160
PatchMatch + Shadows	0.0402	0.2143	0.2282	**0.0351**	0.0969	0.1025
HiFill + Shadows	0.0461	0.2193	0.2346	0.0365	0.0855	**0.0993**
Pix2Pix (all)	0.3583	0.3243	0.4477	0.2502	0.2649	0.3129
Pix2Pix (receiver)	0.0820	0.2118	0.2323	0.1091	0.1526	0.1266
Pix2Pix + Proxy (receiver)	0.0802	0.1766	0.2244	0.0872	0.1187	0.1153
+Intrinsic Decomposition	0.0362	0.0882	0.2238	0.0618	0.0843	0.1055
+Shadow Segmentation	**0.0246**	**0.0713**	0.2213	**0.0340**	0.0631	0.1040
+Lighting Inpainting (Ours)	**0.0248**	**0.0712**	**0.2143**	**0.0340**	**0.0616**	**0.0983**

Table 1: Comparison of error rates for various shadow removal methods

Finally, we impose a sparse gradient prior on L to discourage textural details from leaking into the lighting.

$$E_{\nabla L} = ||\nabla L||_1 \tag{16}$$

Shadow Removal and Lighting Inpainting: As with the intrinsic decomposition, we apply a multiscale loss on the predicted lighting after shadow removal and inpainting.

$$E_{L'} = \lambda_{L'} P(L', \hat{L}') \tag{17}$$

Note that because L' is a composite of the results of the shadow removal and lighting inpainting networks, the shadow removal network is only penalized for pixels lying on the receiver in the original input image while the lighting inpainting network is only penalized for pixels within the mask of the removed object.

We also add a recomposition loss on the final output.

$$E_{I'} = \lambda_{I'} ||\hat{I}' - L'T'||_1 \tag{18}$$

5. Results

We evaluate our work both qualitatively and quantitatively on 5000 synthetic test scenes, generated the same way as our training data, and 14 real scenes captured manually, which include ground truth object removal results.

Commonly used quantitative metrics such as perceptual losses [38, 18] or the RMSE, computed across the entire image, are poor representations of the quality of shadow removal results. A perceptually negligible color cast produced by a deep network across the entire image has an outsized effect on these metrics. To better represent the performance of various systems, we analyze accuracy in targeted regions of the image using a weighted RMSE. In addition to the RMSE across the entire image, we report the Shadow RMSE, which is computed across pixels within the ground truth binary shadow mask $\hat{S}$. We also report the RMSE within the removed object's pixels to separately evaluate inpainting performance.

5.1. Test Data

The proxy geometry and images for real scenes were captured using the Kinect v2 mounted on a tripod. We captured three RGBD frames for each scene: I the complete scene, $\hat{I}'$ the target image with one or more objects removed, and a bare scene with all objects removed. The approximate lighting was captured using a Ricoh Theta S 360 camera with 5 exposures for HDR placed approximately in the center of the scene, roughly pointed at the Kinect.

We median-filtered the depth images to remove noise. We then computed the best-fit plane for the input depth image using RANSAC [12], and computed the receiver mask M_r from pixels approximately lying on the plane. In this work we manually specified the object masks M_o; real applications would use an automatic segmentation method.

To compute the proxy geometry, we formed a triangle mesh from the depth map as we did with the synthetic data; however for cleaner shadows we replaced the vertices corresponding to the ground plane with a fitted plane. This geometry was then rendered with the captured HDR environment lighting to produce P, P'. As we did not have ground truth intrinsic decompositions, we used the difference between I and $\hat{I}'$ instead of $\hat{L}$ and $\hat{L}'$ to produce the shadow mask $\hat{S}$ used in the Shadow RMSE metric.

5.2. End-to-End Comparisons

Most existing works on object removal do not focus on removing shadows cast by the object. We compare to two general approaches as baselines: pure inpainting and generic image-to-image translation. We also include the numerical error of the "no-op" procedure, which does not transform the input at all. Quantitative results are shown in Table 1 and qualitative results in Figure 3, exhibiting varying lighting conditions (multiple overlapping shadows, soft and hard shadows, high and low contrast shadows) and background textures in both synthetic and real test scenes.

We compute the inpainting baselines using two methods: the classical nonparametric PatchMatch [1], and a recent

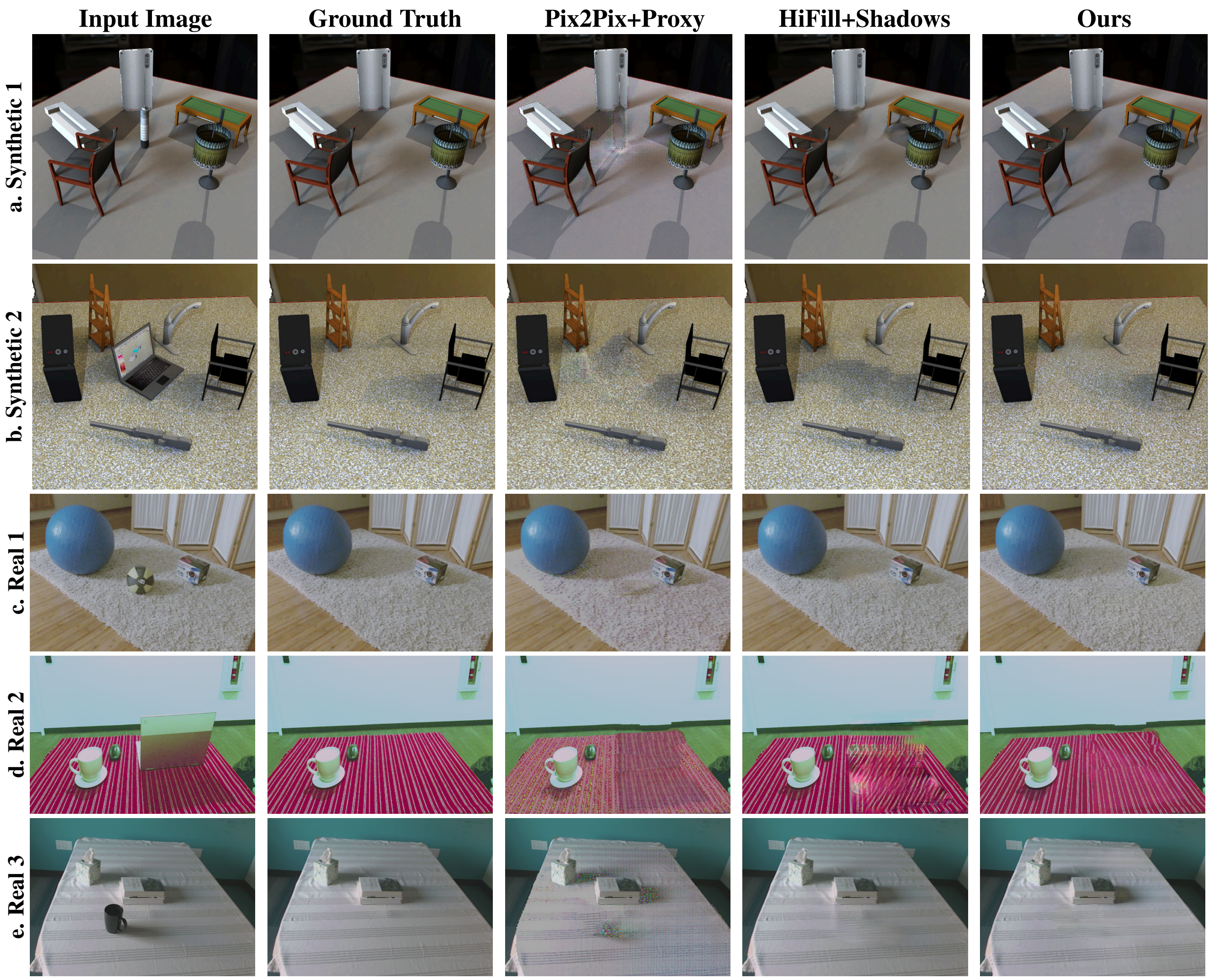


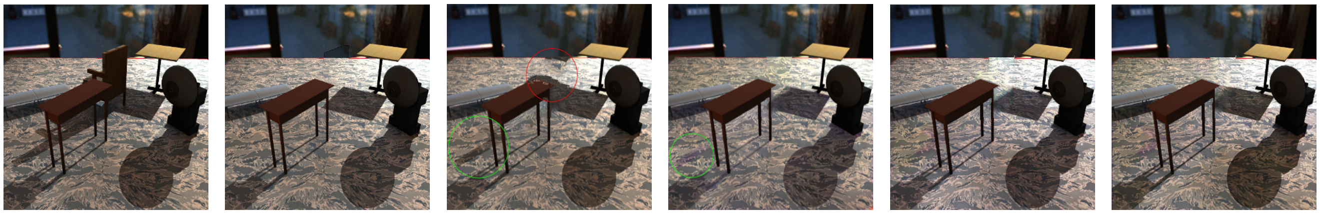
Figure 3: We compare our system to two object removal baselines on both synthetic and real test images. The first baseline is an image-to-image translation network based on Pix2Pix [17] which is supplied with our renderings of the proxy scene. The second baseline is HiFill [35], a state-of-the-art inpainting method, that inpaints both the removed object and an explicitly specified shadow mask.

learning-based approach HiFill [35]. For both baselines, we include quantitative results both for a naive hole-fill, where the object's shadows are not handled at all, as well as for an inpainting mask which includes the object's shadows. In Figure 3 we show the results of HiFill inpainting when provided the *ground truth* shadow region. With this extra information, inpainting approaches work well for simple textures (3a,3c), but often hallucinate shadows within the inpainted region (3b). They also fail to take advantage of texture detail inside the shadow region, and the resulting artifacts are compounded when the region to inpaint is large (3d). Note that, compared to our method, these inpainting baselines do not use the proxy renderings.

For our image-to-image translation baseline, we use the well-known Pix2Pix method [17]. We compare against three variants of this baseline: one trained to predict the entire output image I' from the input I and object mask M_o; one trained to predict only the appearance of the receiving surface (using HiFill to inpaint the object region); and one trained to predict only the appearance of the receiver but supplied with our proxy scene renderings P, P' in addition to the input image and object mask. We show the results of this last version in Figure 3. The method fails to accurately identify the extents of shadows (3c,3e) and their intensities (3a) and generalizes poorly to complex textures (3b,3d).

5.3. Validating our Architecture

We show the importance of each step of our pipeline, starting with a single network to perform our generalized differential rendering task and adding in each component one by one. The results are shown in Figure 4 and the bottom rows of Table 1. We start with a single Pix2pix U-Net



(a) Input (b) Ground Truth (c) Pix2pix (e.g. U-net) (d) +Intrinsic Decomp. (e) +Shadow Segmentation (f) +Lighting Inpainting

Figure 4: This synthetic example (a) shows the importance of each component of our system. The ground truth removal is shown in (b). A single U-net (c) hallucinates shadows in the inpainted region (red) and misidentifies shadows (green) with high-contrast textures. Adding in a texture decomposition subnetwork significantly improves the inpainting (d); however the edges of the shadows are still faintly visible (green). The shadow segmentation subnetwork eliminates these shadow ghosts (e). Some final artifacts visible within the silhouette of the removed object are fixed with our lighting inpainting network (f).

generator, that given all our inputs, predicts the output image excluding the pixels under the object mask, which are inpainted using HiFill (Figure 4c). The most obvious artifacts are within the inpainted region, where the inpainting method frequently fills in shadow pixels; this method also frequently misidentifies shadows, especially in high-contrast textures. We then introduce a separate intrinsic decomposition network, and allow the shadow removal network to work only on the resulting lighting image (Figure 4d); This system sometimes fails to remove hard or high-contrast shadows. Adding a shadow segmentation network (Figure 4e) makes decompositions of hard shadows much cleaner. Finally, we introduce a lighting inpainting network (Figure 4f), as the shadow removal network alone has trouble continuing shadows behind the object and sometimes leaves visible artifacts in the hole region.

Figure 5: Input images (left) virtually refurnished (right).

5.4. Discussion and Future Work

Our shadow removal enables more realistic mixed reality experiences, ranging from consumer applications in real estate and furniture retail, to socially beneficial uses for understanding physical resource allocation in environments such as hospitals and schools. As an example, Figures 1 and 5 show results for a refurnishing scenario. In these results, we run our pipeline twice – once for the wall, and once for the floor. We then composite the two results together, and then insert a virtual object with differential rendering.

In the second row of Figure 5, the glossy hardwood floor shows a specularity of the couch we wish to remove; by adding a specular component to the proxy model's floor plane, our pipeline is able to remove the specularity as well as the shadow. Of course, this specularity is fairly simple, where the couch "occludes" the reflection of the much brighter white wall. Since glossy surfaces are present in many indoor scenes, handling more complex specularities is an important area for further investigation. In this vein, handling higher order light transport effects, such as color bleeding, is also important for realistic results.

Several other artifacts are visible in Figure 5. A faint outline on the left side of the removed couch can be seen; this is due to the light from the window casting nonuniform illumination onto the wall, which was not represented in the training data. The baseboard is also partially removed, even in regions where it was visible in the input image near the piano leg. We built into the training data the assumption that the texture is consistent over the entire receiving surface. Without this restriction, the intrinsic decomposition network tends to incorrectly assign shading variation to the texture image. For similar reasons, we did not consider the shadows cast on other the objects in the scene and focused on a single receiver. Extending the pipeline to handle multiple receivers, or using different representations of textures, would be important additions to our work.

As a final note, any digital image manipulation method carries the risk of misuse. This is especially true with the current prevalence of social media, where false images may be used to spread misinformation and disinformation widely and rapidly. We strongly believe in the importance of research on watermarking and other methods to verify the authenticity of images and track image manipulations.

References

[1] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. In *SIGGRAPH*, 2009. 2, 6

[2] Jonathan T. Barron and Jitendra Malik. Intrinsic scene properties from a single rgb-d image. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2013. 2

[3] Jonathan T. Barron and Jitendra Malik. Shape, illumination, and reflectance from shading. In *IEEE Trans. Pattern Anal. Mach. Intell.*, 2015. 2

[4] Matt Bell and ET Freeman. Learning local evidence for shading and reflectance. In *Int. Conf. Comput. Vis.*, 2001. 2

[5] Sean Bell, Kavita Bala, and Noah Snavely. Intrinsic images in the wild. In *SIGGRAPH*, 2014. 2

[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], 2015. 3

[7] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2014. 3

[8] Angela Dai, Matthias Nießner, Michael Zollöfer, Shahram Izadi, and Christian Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface re-integration. *ACM Trans. Graph.*, 2017. 2

[9] Paul Debevec. Rendering synthetic objects into real scenes. In *SIGGRAPH*, 1998. 1, 2

[10] Bin Ding, Chengjiang Long, Ling Zhang, and Chunxia Xiao. Argan: Attentive recurrent generative adversarial network for shadow detection and removal. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 2

[11] Graham D Finlayson, Steven D Hordley, Cheng Lu, and Mark S Drew. On the removal of shadows from images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(1), 2005. 2

[12] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6), 1981. 6

[13] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2017. 2

[14] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth prediction. In *Int. Conf. Comput. Vis.*, 2019. 2

[15] Roger Grosse, Micah K Johnson, Edward H Adelson, and William T Freeman. Ground truth dataset and baseline evaluations for intrinsic image algorithms. In *Int. Conf. Comput. Vis.*, 2009. 2

[16] Xiaowei Hu, Chi-Wing Fu, Lei Zhu, Jing Qin, and Pheng-Ann Heng. Direction-aware spatial context features for shadow detection and removal. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2019. 2, 4

[17] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 7

[18] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Eur. Conf. Comput. Vis.*, 2016. 6

[19] K Karsch, K. Sunkavalli, S. Hadap, N. Carr, H. Jin, R. Fonte, M. Sittig, and D. Forsyth. Automatic scene inference for 3d object compositing. In *SIGGRAPH*, 2014. 2

[20] Natasha Kholgade, Tomas Simon, Alexei Efros, and Yaser Sheikh. 3d object manipulation in a single photograph using stock 3d models. In *SIGGRAPH*, 2014. 2

[21] Hieu Le and Dimitris Samaras. Shadow removal via shadow image decomposition. In *Int. Conf. Comput. Vis.*, 2019. 2

[22] Chloe LeGendre, Wan-Chun Ma, Graham Fyffe, John Flynn, Laurent Charbonnel, Jay Busch, and Paul Debevec. Deeplight: Learning illumination for unconstrained mobile mixed reality. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 2

[23] Ricardo Martin-Brualla, Rohit Pandey, Shuoran Yang, Pavel Pidlypenskyi, Jonathan Taylor, Julien Valentin, Sameh Khamis, Philip Davidson, Anastasia Tkach, Peter Lincoln, Adarsh Kowdle, Christoph Rhemann, Dan B Goldman, Cem Keskin, Steve Seitz, Shahram Izadi, and Sean Fanello. Lookingood: Enhancing performance capture with real-time neural re-rendering. In *SIGGRAPH Asia*, 2018. 2

[24] Moustafa Meshry, Dan B. Goldman, Sameh Khamis, Hugues Hoppe, Rohit Pandey, Noah Snavely, and Ricardo Martin-Brualla. Neural rerendering in the wild. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 2

[25] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically Based Rendering: From Theory to Implementation*. Morgan Kaufmann Publishers Inc., San Francisco, USA, 3rd edition, 2016. 3

[26] Julien Philip, Michaël Gharbi, Tinghui Zhou, Alexei A. Efros, and George Drettakis. Multi-view relighting using a geometry-aware network. In *SIGGRAPH*, 2019. 2

[27] Liangqiong Qu, Jiandong Tian, Shengfeng He, Yandong Tang, and Rynson WH Lau. Deshadownet: A multi-context embedding deep network for shadow removal. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 2

[28] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 4

[29] Shuran Song and Thomas Funkhouser. Neural illumination: Lighting prediction for indoor environments. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 2

[30] Pratul P. Srinivasan, Ben Mildenhall, Matthew Tancik, Jonathan T. Barron, Richard Tucker, and Noah Snavely. Lighthouse: Predicting lighting volumes for spatially-coherent illumination. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 2

[31] Marshall F Tappen, William T Freeman, and Edward H Adelson. Recovering intrinsic images from a single image. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2005. 2

[32] Kimmo Valkealahti and Erkki Oja. Reduced multidimensional co-occurrence histograms in texture classification. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(1), 1998. 3

[33] Jifeng Wang, Xiang Li, and Jian Yang. Stacked conditional generative adversarial networks for jointly learning shadow detection and shadow removal. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. 2

[34] Tianyu Wang, Xiaowei Hu, Qiong Wang, Pheng-Ann Heng, and Chi-Wing Fu. Instance shadow detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 2

[35] Zili Yi, Qiang Tang, Shekoofeh Azizi, Daesik Jang, and Zhan Xu. Contextual residual aggregation for ultra high-resolution image inpainting. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 2, 5, 7

[36] Yu Zeng, Zhe Lin, Jimei Yang, Jianming Zhang, Eli Shechtman, and Huchuan Lu. High-resolution image inpainting with iterative confidence feedback and guided upsampling. In *Eur. Conf. Comput. Vis.*, 2020. 2

[37] Edward Zhang, Michael F. Cohen, and Brian Curless. Emptying, refurnishing, and relighting indoor spaces. In *SIGGRAPH Asia*, 2016. 2

[38] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. 6

[39] Xuaner Zhang, Ren Ng, and Qifeng Chen. Single image reflection separation with perceptual losses. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. 5

[40] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2017. 5