Neural Inverse Rendering of an Indoor Scene From a Single Image

Soumyadip Sengupta[1,2,3,†], Jinwei Gu[1,4,†], Kihwan Kim[1], Guilin Liu[1], David W. Jacobs[2], and Jan Kautz[1]

[1]NVIDIA, [2]University of Maryland, College Park, [3]University of Washington, [4]SenseTime

1. Overview

In this supplementary material, we provide more details of our network architecture and the loss functions along with additional qualitative evaluations. Specifically, in Section 2 we discuss the details of the IRN and RAR network architectures for reproducibility. Details of our training loss functions on real data are provided in Section 3. In Section 5 we present additional qualitative evaluations.

2. Network Architectures

Our proposed Inverse Rendering Network (IRN), shown again in Figure 1 for reference, is trained on real data using the Residual Appearance Renderer (RAR), which learns to capture the complex appearance effects (*e.g.* inter-reflection, cast shadows, near-field illumination, and realistic shading). In the following, we describe the implementation details of IRN and RAR.

2.1. IRN

In Figure 2 we present the network architecture of IRN. The input to IRN is an image of spatial resolution 240×320, and the output is an albedo and normal map of same spatial resolution along with a 18×36 resolution environment map. Here we provide the details of the each block in IRN.

'Enc': C64(k7) - C*128(k3) - C*256(k3)
'CN(kS)' denotes convolution layers with N $S \times S$ filters with stride 1, followed by Batch Normalization and ReLU. 'C*N(kS)' denotes convolution layers with N $S \times S$ filters with stride 2, followed by Batch Normalization and ReLU. The output of 'Enc' layer produces a blob of spatial resolution $256 \times 60 \times 80$.

'Normal ResBLKs': 9 ResBLK
This consists of 9 Residual Blocks, 'ResBLK's, which operate at a spatial resolution of $256 \times 60 \times 80$. Each 'ResBLK' consists of Conv256(k3) - BN -ReLU - Conv256(k3) - BN, where 'ConvN(kS)' and 'BN' denote convolution layers with N $S \times S$ filters of stride 1 and Batch Normalization.
'Albedo ResBLKs': Same as 'Normal Residual Blocks' (weights are not shared).

'Dec.': CD*128(k3)-CD*64(k3)-Co3(k7)
'CD*N(kS)' denotes Transposed Convolution layers with N $S \times S$ filters with stride 2, followed by Batch Normalization and ReLU. 'CN(kS)' denotes convolution layers with N $S \times S$ filters with stride 1, followed by Batch Normalization and ReLU. The last layer Co3k(7) consists of only convolution layers of 3 7×7 filters, followed by Tanh layer.

'Light Est.': It first concatenates the responses of 'Enc', 'Normal ResBLKs' and 'Albedo ResBLKs' to produce a blob of spatial resolution $768 \times 60 \times 80$. It is further processed by the following module:
C256(k1) - C*256(k3) - C*128(k3) - C*3(k3) - BU(18,36)
'CN(kS)' denotes convolution layers with N $S \times S$ filters with stride 1, followed by Batch Normalization and ReLU. 'C*N(kS)' denotes convolution layers with N $S \times S$ filters with stride 2, followed by Batch Normalization and ReLU. BU(18,36) upsamples the response to produce $18 \times 36 \times 3$ resolution environment map.

2.2. RAR

As shown in Figure 1, Residual Appearance Renderer (RAR) consists of a U-Net architecture and a convolution encoder. The U-Net consists of the following architecture, with normals and albedo as its input:
'Encoder': C64(k3) - C*64(k3) - C*128(k3) - C*256(k3) - C*512(k3)
'Decoder': CU512(k3) - CU256(k3) - CU128(k3) - CU64(k3) - Co3(k1)
'CN(kS)' denotes convolution layers with N $S \times S$ filters with stride 1, followed by Batch Normalization and ReLU. 'C*N(kS)' denotes convolution layers with N $S \times S$ filters with stride 2, followed by Batch Normalization and ReLU. 'CUN(kS)' represents a bilinear up-sampling layer , followed by convolution layers with N $S \times S$ filters with stride 1, followed by Batch Normalization and ReLU. 'Co3(k1)' consists of 3 1×1 convolution filters to produce Normal or

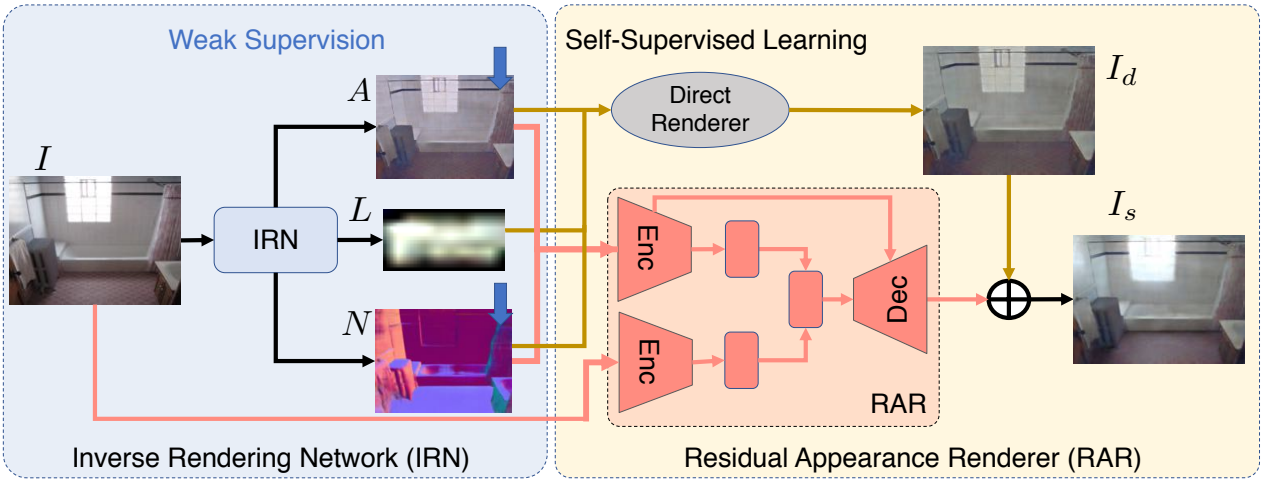

Figure 1: **Our Proposed Architecture**.

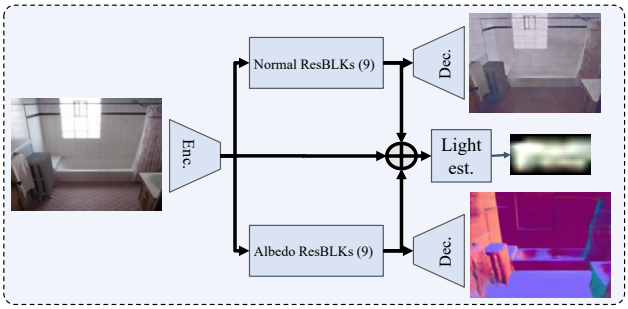

Figure 2: **IRN**.

Albedo. Skip-connections exists between 'C*N(k3)' layers of encoder and 'CUN(k3)' layers of decoder. The encoder 'Enc' that encodes image features to a latent $D = 300$ dimensional subspace is given by: 'Enc': C64(k7) - C*128(k3) - C*256(k3) - C128(k1) - C64(k3) - C*32(k3) - C*16(k3) - MLP(300)
'CN(kS)' denotes convolution layers with N $S \times S$ filters with stride 1, followed by Batch Normalization and ReLU. 'C*N(kS)' denotes convolution layers with N $S \times S$ filters with stride 2, followed by Batch Normalization and ReLU. MLP(300) takes the response of the previous layer and outputs a 300 dimensional feature, which is concatenated with the last layer of the U-Net 'Encoder'.

2.3. Environment Map Estimator

As discussed in Section 3.1 of the main paper, the ground-truth environment map is estimated from the image, ground-truth albedo and normal using a deep network $h_e(\cdot, \Theta_e)$. The detailed architecture of this network is presented below:
C64(k7) - C*128(k3) - C*256(k3) - 4 ResBLKS - C256(k1) - C*256(k3) - C*128(k3) - C*3(k3) - BU(18,36),
where, 'CN(kS)' denotes convolution layers with N $S \times S$ filters with stride 1, followed by Batch Normalization and ReLU. 'C*N(kS)' denotes convolution layers with N $S \times S$ filters with stride 2, followed by Batch Normalization and ReLU. BU(18,36) upsamples the response to produce $18 \times 36 \times 3$ resolution environment map. Each 'ResBLK' contains Conv256(k3) - BN -ReLU - Conv256(k3) - BN, where 'ConvN(kS)' denotes convolution layers with N $S \times S$ filters of stride 1, 'BN' denoted Batch Normalization.

3. Training Details

3.1. Training with weak supervision over albedo

IIW dataset presents relative reflectance judgments from humans. For any two points R_1 and R_2 on an image, a weighted confidence score classifies R_1 to be same, brighter or darker than R_2. We use these labels to construct a hinge loss for sparse supervision based on WHRD metric presented in [2]. Specifically, if users predict R_1 to be darker than R_2 with confidence w_t, we use a loss $w_t \max(1 + \delta - R_2/R_1, 0)$. If R_1 and R_2 are predicted to have similar reflectance, we use $w_t[\max(R_1/R_2 - 1 - \delta, 0) + \max(R_2/R_1 - 1 - \delta, 0)]$. We observed empirically that this loss function performs better than WHRD metric, which is an L0 version of our loss. We train on real data with the following losses: (i) Psuedo-supervision loss over albedo (L_a), normal (L_n) and lighting (L_e) based on [6], (ii) Photometric Reconstruction loss with RAR (L_u) (iii) Pairwise weak supervision (L_w). Thus the net loss function is defined as:

$$L = 0.5 * L_a + 0.5 * L_n + 0.1 * L_e + L_u + 30 * L_w. \quad (1)$$

3.2. Training with weak supervision over normals

We also train on NYUv2 dataset with weak supervision over normals, obtained from Kinect depth data of the scene. We train with the following losses: (i) Psuedo-supervision

loss over albedo (L_a) and lighting (L_e) based on [6], (ii) Photometric Reconstruction loss with RAR (L_u) (iii) Supervision (L_w) over kinect normals. Thus the net loss function is defined as:

$$L = 0.2 * L_a + 0.05 * L_e + L_u + 20 * L_w. \quad (2)$$

4. Our CG-PBR Dataset

We present more example images of our CG-PBRS dataset in Figure 3. We also compare the renderings of our CG-PBR Dataset with that of PBRS [7], under the same illumination condition in Figure 4 and 5. CG-PBR provides more photo-realistic and less noisy images with specular highlights. Both CG-PBR and PBRS are rendered with Mitsuba [4].

5. More Experimental Results

Comparison with SIRFS. We present more detailed qualitative evaluations in this supplementary material. In Figure 6 we compare the results of our algorithm with that of SIRFS [1]. SIRFS is an optimization-based method for inverse rendering, which estimates surface normals, albedo and spherical harmonics lighting from a single image. Compared to SIRFS we obtain more accurate normals and better disambiguation of reflectance from shading.

Comparison with Li *et al*. [5]. In Figure 7 and 8 we compare the albedo predicted by our method with that of Li *et al*. [5], which performs intrinsic image decomposition of an image, on the real IIW and the synthetic CG-PBR dataset respectively. Intrinsic image decomposition methods do not explicitly recover geometry, illumination or glossiness of the material, but rather combine them together as shading. In contrast, our goal is to perform a complete inverse rendering which has a wider range of applications in AR/VR.

Evaluation of lighting estimation. In Figure 9 we present a qualitative evaluation of lighting estimation by inserting a diffuse hemisphere into the scene and rendering it with the inferred light from the image. We compare this with the method proposed by Gardner *et al*. [3], which also estimates an environment map from a single indoor image. $h_e(\cdot, \Theta_e)$ is a deep network that predicts the environment map given the image, normals, and albedo. 'GT+$h_e(\cdot)$' estimates the environment map given the image, ground-truth normals and albedo, and thus serves as an achievable upper-bound in the quality of the estimated lighting. 'Ours' estimates environment map from an image with IRN. 'Ours+$h_e(\cdot)$' predicts environment map by combining the inferred albedo and normals from IRN to predict lighting with $h_e(\cdot)$. Both 'Ours' and 'Ours+$h_e(\cdot)$' outperform Gardner *et al*. [3] as they seem to produce more realistic environment maps. 'Ours+$h_e(\cdot)$' improves lighting estimation over 'Ours' by utilizing the predicted albedo and normals to a greater degree.

Our Results and Ablation study. Figure 10 shows examples of our results, with the albedo, normal and lighting predicted by the network, as well as the reconstructed image with the direct renderer and the proposed Residual Appearance Renderer (RAR). In Figure 11 and 12, we perform a detailed ablation study of different components of our method. We show that it is important to train on real data, as networks trained on synthetic data fail to generalize well on real data. We also show that training our method without RAR ('w/o RAR') produces piece-wise smooth, low contrast albedo due to over-reliance on the weak supervision of pair-wise relative reflectance judgments. Training our network with only RAR, without any weak supervision ('w/o weak'), often fails to produce consistent albedo across large objects like walls, floor, ceilings etc. Finally, training without RAR and weak supervision ('w/o weak + RAR') produces albedo which contains the complex appearance effects like cast shadows, inter-reflections, highlights etc., as the reconstruction loss with direct renderer alone cannot model these effects.

References

[1] Jonathan T Barron and Jitendra Malik. Shape, illumination, and reflectance from shading. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2015. 3, 6

[2] Sean Bell, Kavita Bala, and Noah Snavely. Intrinsic images in the wild. *ACM Transactions on Graphics (TOG)*, 33(4):159, 2014. 2

[3] Marc-André Gardner, Kalyan Sunkavalli, Ersin Yumer, Xiaohui Shen, Emiliano Gambaretto, Christian Gagné, and Jean-François Lalonde. Learning to predict indoor illumination from a single image. *ACM Transactions on Graphics (TOG)*, 36(6):176, 2017. 3, 8

[4] Wenzel Jakob. Mitsuba renderer, 2010. http://www.mitsuba-renderer.org. 3

[5] Zhengqi Li and Noah Snavely. CGIntrinsics: Better intrinsic image decomposition through physically-based rendering. *European Conference on Computer Vision (ECCV)*, 2018. 3, 7

[6] Soumyadip Sengupta, Angjoo Kanazawa, Carlos D. Castillo, and David W. Jacobs. Sfsnet: Learning shape, refectance and illuminance of faces in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 3

[7] Yinda Zhang, Shuran Song, Ersin Yumer, Manolis Savva, Joon-Young Lee, Hailin Jin, and Thomas Funkhouser. Physically-based rendering for indoor scene understanding using convolutional neural networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3, 5

Figure 3: **Our CG-PBR Dataset** concists of 235,893 images of a scene assuming specular and diffuse reflectance along with ground truth depth, surface normals, albedo, Phong model parameters, semantic segmentation and glossiness segmentation.

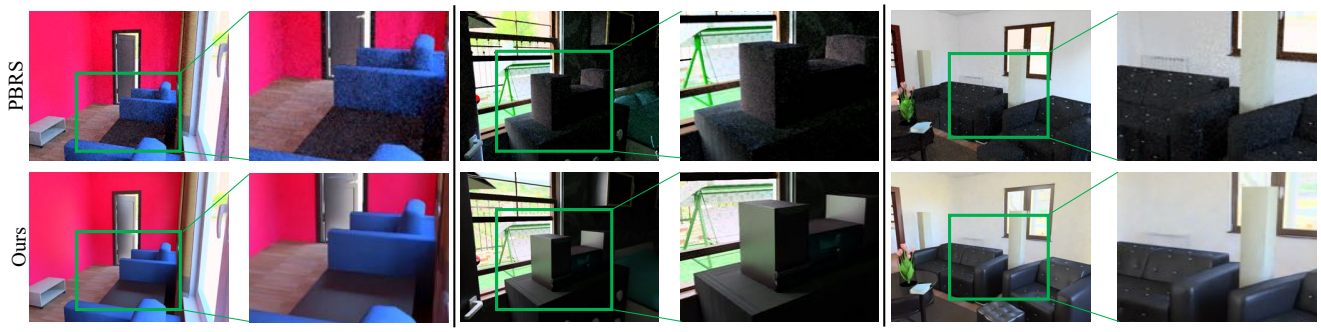

Figure 4: **Comparison with PBRS [7].** Our dataset provides more photo-realistic and less noisy images with specular highlights under multiple lighting conditions.

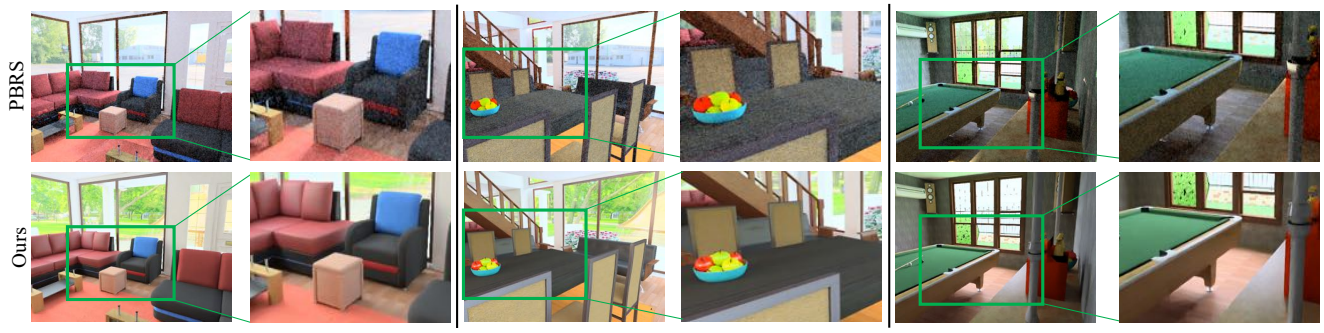

Figure 5: **Comparison with PBRS [7].** Our dataset provides more photo-realistic and less noisy images with specular highlights under multiple lighting conditions.

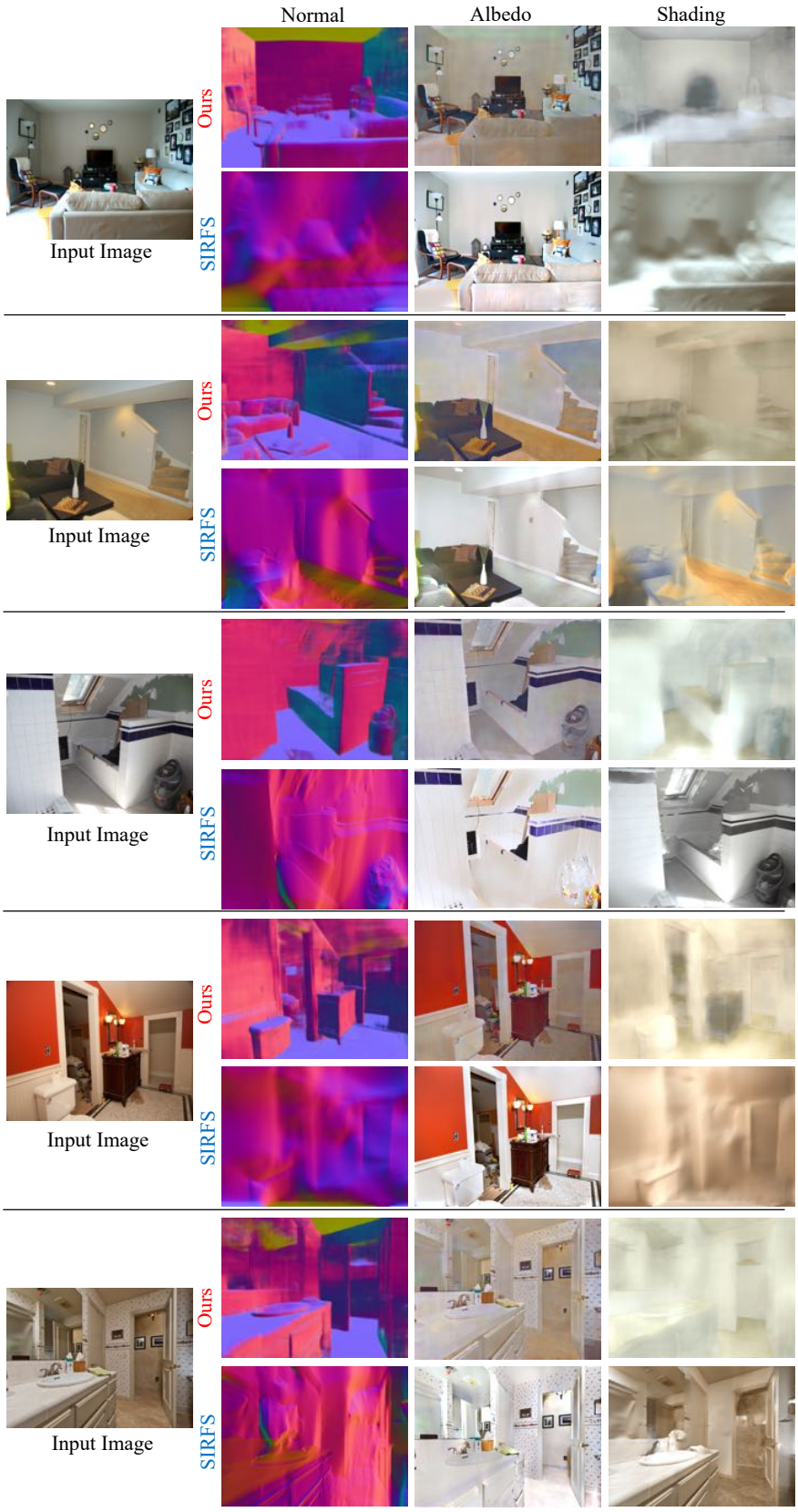

Figure 6: **Comparison with SIRFS [1].** Using deep CNNs our method performs better disambiguation of reflectance from shading and predicts better surface normals.

Figure 7: **Comparison with CGI (Li *et. al.* [5]) on IIW dataset.** In comparison with Li *et al*. [5], our method performs better disambiguation of reflectance from shading and preserves the texture in the albedo.

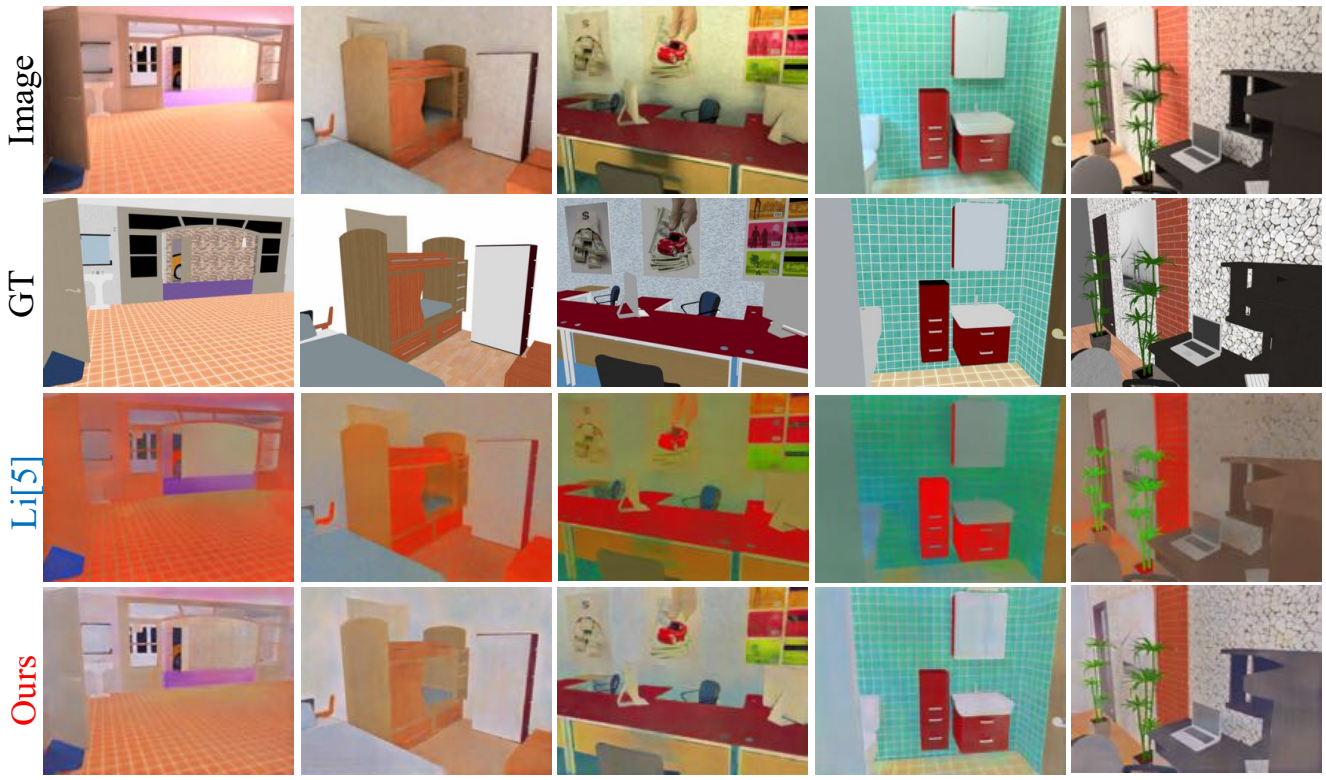

Figure 8: **Comparison with CGI (Li *et. al.* [5]) on CG-PBR (synthetic) dataset.** In comparison with Li *et al*. [5], our method performs better disambiguation of reflectance from shading and preserves the texture in the albedo.

Figure 9: **Evaluation of lighting estimation.** We compare with Gardner *et al*. [3]. 'GT+$h_e(\cdot)$' predicts lighting conditioned on the ground-truth normals and albedo. 'Ours+$h_e(\cdot)$' predicts the environment map by conditioning it on the albedo and normals inferred by IRN.

Figure 10: **Our Result.** We show the estimated intrinsic components; normals, albedo, and lighting predicted by the network, along with the reconstructed image with our direct renderer and the RAR.

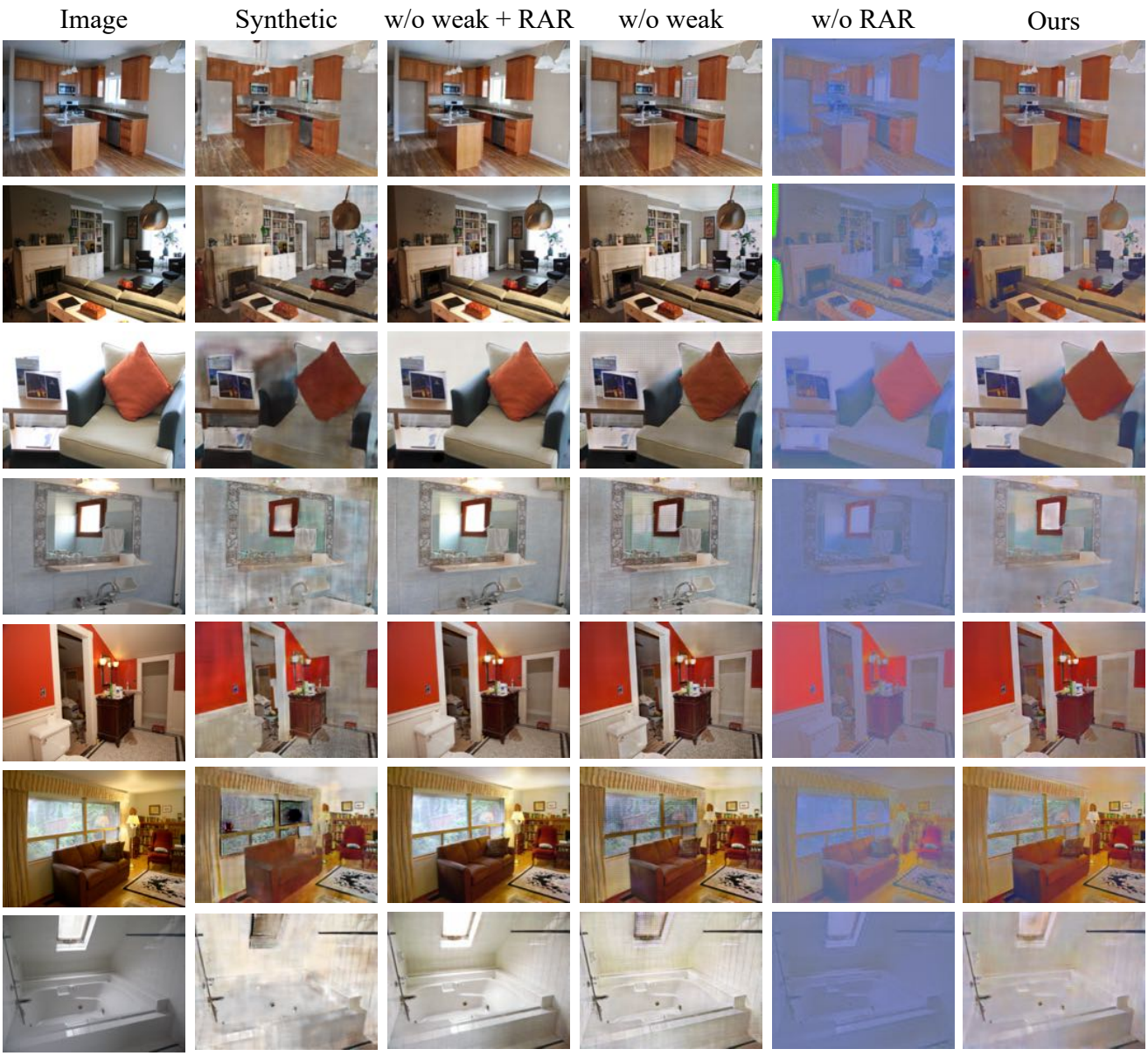

Figure 11: **Ablation Study**. We present the predicted albedo for each input image (in column 1) in column 2-6. We show the albedo predicted by IRN trained on our CG-PBR only in column 2. In column 6 ('Ours') we show the albedo predicted by IRN finetuned on real data with RAR and weak supervision over albedo. In column 4 and 5 we show the albedo predicted by IRN finetuned on real data without weak supervision ('w/o weak') and RAR ('w/o RAR') respectively. We present the albedo predicted by IRN finetuned without RAR and weak supervision on real data in column 3 ('w/o weak + RAR'). More images in Figure 12.

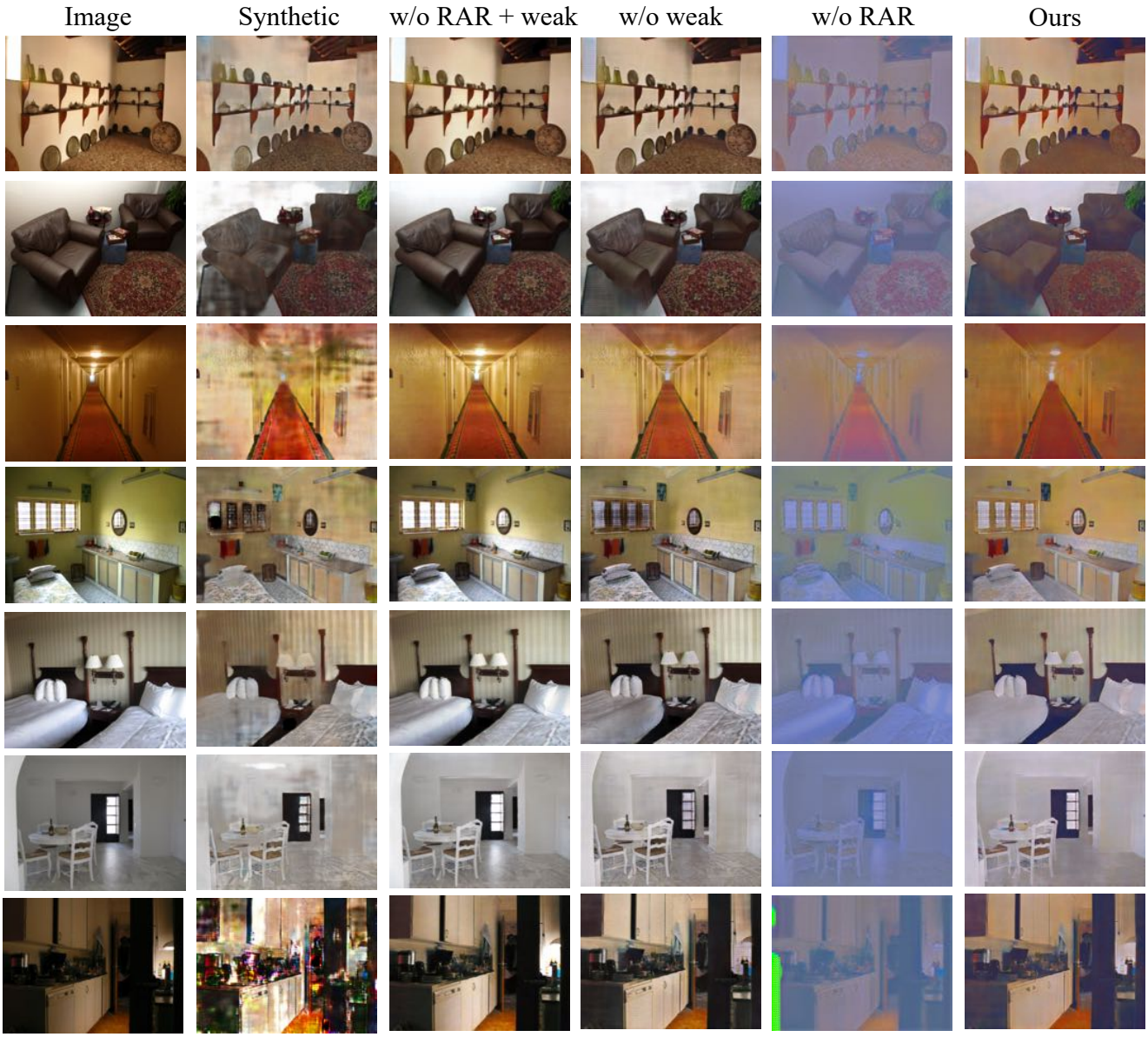

Figure 12: **Ablation Study**. We present the predicted albedo for each input image (in column 1) in column 2-6. We show the albedo predicted by IRN trained on our CG-PBR only in column 2. In column 6 ('Ours') we show the albedo predicted by IRN finetuned on real data with RAR and weak supervision over albedo. In column 4 and 5 we show the albedo predicted by IRN finetuned on real data without weak supervision ('w/o weak') and RAR ('w/o RAR') respectively. We present the albedo predicted by IRN finetuned without RAR and weak supervision on real data in column 3 ('w/o weak + RAR').