Estimating Spatially-Varying Lighting in Urban Scenes with Disentangled Representation

Jiajun Tang[1], Yongjie Zhu[2], Haoyu Wang[1], Jun Hoong Chan[1], Si Li[2], and Boxin Shi[1,3] ✉

[1] NERCVT, School of Computer Science, Peking University
[2] School of Artificial Intelligence, Beijing University of Posts and Telecommunications
[3] Peng Cheng Laboratory

Abstract. We present an end-to-end network for spatially-varying outdoor lighting estimation in urban scenes given a single limited field-of-view LDR image and any assigned 2D pixel position. We use three disentangled latent spaces learned by our network to represent sky light, sun light, and lighting-independent local contents respectively. At inference time, our lighting estimation network can run efficiently in an end-to-end manner by merging the global lighting and the local appearance rendered by the local appearance renderer with the predicted local silhouette. We enhance an existing synthetic dataset with more realistic material models and diverse lighting conditions for more effective training. We also capture the first real dataset with HDR labels for evaluating spatially-varying outdoor lighting estimation. Experiments on both synthetic and real datasets show that our method achieves state-of-the-art performance with more flexible editability.

Keywords: Spatially-varying lighting; Disentangled representation; Lighting estimation; Urban scenes

1 Introduction

Single image based outdoor illumination estimation, aiming at estimating lighting from a single limited field-of-view (FoV) image, takes a crucial role in many computer vision applications, such as object relighting, scene understanding, and augmented reality (AR). Unlike indoor scenarios, outdoor lighting contains rich high-frequency and high-intensity components. In early works, several low-dimensional parametric models are proposed to fit a distant global lighting, such as the Hošek-Wilkie (HW) sky model [13,14] and the Lalonde-Matthews (LM) sky model [17]. However, the capacities of those parametric models are not sufficient for real-world outdoor lighting, which often leads to unrealistic rendering results. Recent data-driven approaches start using latent spaces learned by autoencoders to represent outdoor lighting conditions to serve as the target of global lighting estimation [11]. Methods combining merits of latent space representation ability and interpretable parameters have also been proposed [26].

✉ Corresponding author: shiboxin@pku.edu.cn

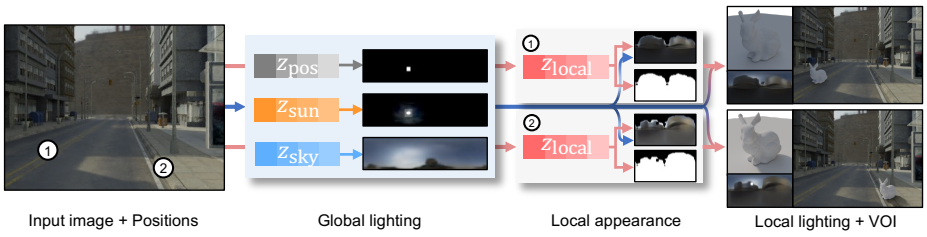


Fig. 1. We propose an end-to-end network for spatially-varying outdoor lighting estimation. Given a limited-FoV LDR image and the 2D pixel positions, the spatially-uniform global lighting is first estimated as the sun position, sun light, and sky light. The spatially-varying local content at each position is then estimated respectively. The spatially-varying local appearance is rendered with both local content information and global lighting condition, and the spatially-varying lighting at each position is finally obtained by merging the local appearance with the global lighting using the predicted local silhouette mask. Realistic virtual object insertion (VOI) can be conducted using the predicted local lighting (panoramas are shown in tone-mapped HDR).

However, these methods all treat the outdoor lighting as a single *spatially-uniform (global)* lighting, *i.e.*, the light probe is surrounded by an environment map that casts rays from infinitely far away. Extending outdoor lighting estimation to support *spatially-varying (local)* prediction is of important practicability, especially for urban scenes where many buildings occlude the light path making the spatially-uniform representation rather unrealistic. Only until very recently, such a problem has been demonstrated to be solvable via intrinsic decomposition and panoramic environment map completion [28].

Although the first trial for outdoor spatially-varying lighting estimation [28] has shown its impressive performance, there still remains room for improvement: 1) The direct usage of panoramic environment maps is a less compact representation compared to latent parametric models [11]. 2) The estimated lightings in the form of panoramic maps have limited editability (horizontal rotation and scaling). 3) The SOLID-Img [28] synthetic dataset are with limited diversity (only default object materials in Blender SceneCity [2] and sunny environment maps) and there lacks a real-captured outdoor spatially-varying dataset with ground truth HDR lightings to more comprehensively evaluate the performance.

In this paper, we propose the *SOLD-Net* for *Spatially-varying Outdoor Lighting estimation with Disentangled representation*, which consists of the end-to-end *spatially-varying lighting estimator* for non-uniform lighting estimation on disentangled latent spaces and the *global lighting encoder-decoder* together with *local content encoder-renderer* for learning spatially-varying lighting representation. As shown in Figure 1, the global lighting encoder-decoder learns to disentangle global lighting as two different latent spaces for sky light and sun light with separate parameters such as the sun position; the lighting-dependent spatially-varying local appearances are rendered given global lighting conditions and the lighting-independent local content information learned by the local

content encoder-renderer. Our method achieves state-of-the-art performance on spatially-varying outdoor lighting estimation with the following contributions:

- using disentangled latent spaces to compactly represent global lighting conditions and local contents;
- designing an end-to-end network architecture to infer spatially-varying lighting with flexible editability;
- enhancing spatially-varying outdoor lighting estimation datasets by increasing material diversity and weather diversity for the synthetic dataset and capturing the first real dataset with HDR ground truth labels.

2 Related Work

Our method targets at lighting estimation from a single image for the outdoor urban scenario with local lighting effects being considered. In this section, we briefly review relevant works for outdoor lighting estimation first and then discuss how existing methods consider local lighting estimation.

Outdoor lighting estimation. The most direct way to obtain outdoor lighting representation is to capture HDR images with multiple exposures that include the sun and sky [25]. However, estimating illumination conditions from images is always desired for practical consideration. This is feasible since an outdoor image provides useful cues that could reveal the surrounding environment. Lalonde *et al.* [15] for the first time infer illumination from shadows, shading, and sky appearance variations observed in the image. A convolutional neural network [7] is used to process the symmetric view of pairwise photos captured from rear and front cameras of mobile devices, and to estimate the outdoor lighting represented by low-frequency spherical harmonic (SH) coefficients. However, it has a difficulty in dealing with high-frequency information. A physics-based Hošek-Wilkie (HW) sky model [13,14] is better tailored to recover HDR parameters for deep outdoor illumination estimation [12], but it is sensitive to cloud patches data. This issue is solved by using a more robust sky model, the Lalonde-Matthews (LM) model [17,27], which covers more comprehensive lighting conditions in the outdoor environment. More recently, a novel SkyNet autoencoder [11] is designed to learn a latent sky model from a large sky panorama dataset [16] and successfully estimates outdoor lighting from limited-FoV images. An encoder-decoder framework is further proposed [18] to learn a mapping from a limited-FoV LDR image to HDR lighting and a mobile phone camera equipped with three spheres of different reflectance is used to capture ground truth training data. HDSky [26] makes several physically meaningful attributes of global outdoor lighting estimation editable by hierarchically training autoencoders with different supervisions.

Local lighting estimation. Different from global illumination representation, the local illumination is related to the position where the light probe is placed, *i.e.*, to capture the lighting intensity at a target position using a mirror sphere in the scene [8]. However, calibrations of objects are not always available, so local lighting estimates from images attract researchers' attention. Earlier works infer

intrinsic properties from a single RGBD image, and a noisy depth image is used to improve spherical harmonic estimation [5], but such an approach does not capture abrupt changes in ambient lighting caused by local geometry. Estimating spatially-varying indoor lighting from a single RGB image could be conducted in real-time [10], by using high-order SH representations learned by deep neural networks. After the first usage of spherical Gaussian (SG) in deep indoor lighting estimation [9], an inverse rendering network [19] is proposed to estimate indoor spatially-varying spherical Gaussian coefficients for scene editing. NeurIllum [23] recovers high-frequency local lighting with warped color image according to recovered geometry, which shows promising texture details, but the lighting estimations are sometimes imprecise due to the errors in the estimated geometry and the light which is not directly observed in the input. Lighthouse [24] achieves a spatially-coherent (varies smoothly in 3D) and spatially-varying indoor lighting estimation from stereo images using the 3D volumetric RGBA lighting model.

Spatially-varying lighting estimation for outdoor scenarios has not been demonstrated until the proposing of SOLID-Net [28], which is the most relevant work to ours. SOLID-Net relies on decomposing scene intrinsics first for global illumination and then 'warp and complete' a panoramic environment map to include local information. Our method uses only a few parameters in disentangled latent spaces to encode sky, sun, and local information compactly, and the local lighting can be estimated in an end-to-end manner at inference time.

3 Problem Formulation

As aforementioned, the outdoor lighting, especially in urban scenes, is spatially-varying and consists of two parts: 1) the *spatially-uniform (global)* part which can be approximately seen as light sources from infinitely far away and doesn't change with the view point, such as lights from far background objects in the sky-dome. 2) the *spatially-varying (local)* part whose changes with the view point are not eligible, mainly the lights come from or occluded by nearby ground, buildings or plants.

Accordingly, when using panoramic environment maps to present lighting, the global part of outdoor lighting corresponds to the global lighting map P_{global}. Since the sun light and the sky light are two different types of light sources, P_{global} can be further approximately decomposed into the ambient lighting from the sky P_{sky} and the distant lighting from the sun P_{sun}:

$$P_{\text{global}} = P_{\text{sky}} + z_{\text{vis}}(P_{\text{sun}} \odot M_{\text{sun}}), \tag{1}$$

where M_{sun} is the binary panoramic mask indicating the position of the sun, and $z_{\text{vis}} \in [0, 1]$ indicates the visibility of the sun under different conditions.

Note that in daytime, natural light is dominant over artificial light, thus the local part of outdoor lighting can be treated as the illuminated local appearance P_{app} by global lights from P_{global}:

$$P_{\text{app}} = \Phi(z_{\text{local}}, P_{\text{global}}), \tag{2}$$

where Φ denotes the lighting (or rendering) process, and z_{local} is the local content information containing *lighting-independent* properties (such as implicit albedo, normal, and silhouette) of forground objects. Here by indicating *local*, we mean *from the panoramic view* centered at the view point (x, y, z) in the 3D space corresponding to the pixel position (u, v) in a limited-FoV image.

Finally, we can get the spatially-varying local lighting map P_{local} by:

$$P_{\text{local}} = P_{\text{global}} \odot (1 - M_{\text{sil}}) + P_{\text{app}} \odot M_{\text{sil}}, \tag{3}$$

where M_{sil} is the panoramic silhouette of local content (foreground objects).

An intuitive illustration of the advantages of our disentangled lighting representation is shown in Figure 2. Spherical harmonics (SH) [21] is ineffective to model high-frequency sun light resulting in soft shadows in the relighting, optimized spherical Gaussian (SG) [19] and our lighting representation both give a realistic relighting, while our lighting representation provides better editability by disentangling different types of lighting (ambient sky light, distant sun light, and illuminated local appearance). Based on this lighting representation, we design an end-to-end network (in Sec. 4) to estimate spatially-uniform global lighting P_{global} and spatially-varying local lighting P_{local} at the same time.

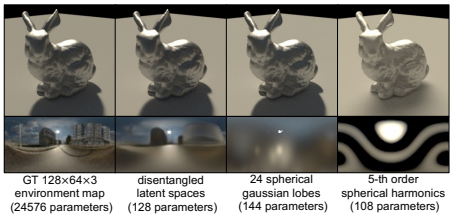


Fig. 2. The relighting results and corresponding environment maps of ground truth, our disentangled lighting representation, optimized spherical Gaussian [19], and spherical harmonics [21]. Our representation gives both realistic relighting and better editability.

4 Method

This section introduces the datasets we use (in Sec. 4.1), the design methodology (in Sec. 4.2), the training (in Sec. 4.3) and inference (in Sec. 4.4) procedures of our method, whose pipeline is shown in Figure 3.

4.1 Dataset

As far as we know, the SOLID-Img dataset [28] is the only suitable one for spatially-varying outdoor lighting estimation. However, its synthetic and real-world counterparts have limitations in terms of material diversity and ground truth quality. We expand both parts as our training and testing datasets.

Synthetic dataset enhancement. The SOLID-Img [28] dataset is rendered using a 3D city model from Blender SceneCity [2] and 70 HDR environment maps for different global lighting conditions and has 38,000 images. It only uses

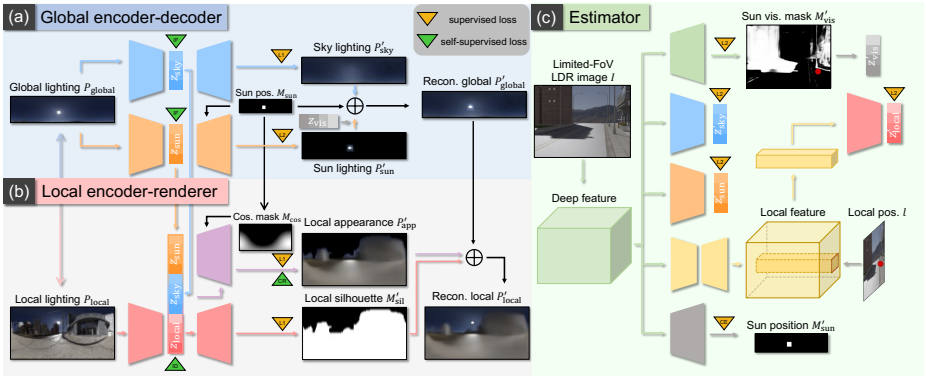


Fig. 3. Overview of our SOLD-Net. Our method uses (a) a global lighting encoder-decoder and (b) a local content encoder-renderer to learn disentangled latent spaces for global (sky, sun) and local information respectively, then (c) a spatially-varying lighting estimator predicts the spatially-varying lighting in disentangled lighting components from a single limited-FoV LDR image with a given pixel position.

12 default materials in Blender SceneCity [2], which are less realistic diffuse materials and might lead to insufficient generalization ability of trained models on real data. To make a dataset with greater diversity, we use a set of 30 open physically based rendering (PBR) materials [1] which contain color maps, normal maps, roughness maps, and displacement maps of the Disney principled BRDF model [6]. As shown in Figure 4, we randomly apply different materials in the 3D city model according to the specific object type, and the issues of repetitive buildings and unrealistic grounds are solved. To cover more diverse lighting conditions, we select 108 representative outdoor HDR environment maps from Poly Haven [3] (including 75 sunny, 15 cloudy, and 18 partly-cloudy panoramas). By this way, we narrow the data gap and make the trained model more robust on real data. Moreover, we render paired local lighting maps of the same local content with different global lighting conditions for our self-supervised cross rendering loss (in Sec. 4.3). Finally, we render 151,632 images in total.

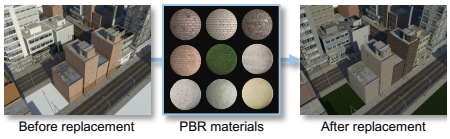


Fig. 4. An example of rendered 3D city model (bird view) before and after we randomly replacing the materials in the model with PBR materials in the same object category.

Real data with HDR local lightings. The real test data in SOLID-Net[28] only have LDR panoramic environment maps as ground truth labels of spatially-varying lighting, which prevents us quantitatively measure the performance and qualitatively compare the relighting results. To provide a more comprehensive

evaluation of outdoor local lighting estimation, we capture a test dataset of real outdoor urban scenes and the corresponding HDR spatially-varying local environment maps. The scenes are captured by a Sony ILCE-7RM3 camera. To obtain unclipped HDR capture of the sun intensity, we first capture a basic HDR panorama using a Ricoh Theta Z1 camera in which only the sun pixels is clipped, and then capture the sun region directly using the Sony ILCE-7RM3 camera equipped with a 3.0 neutral density (ND) filter; the final unclipped HDR panoramic environment maps are stitched by aligning the sun positions in panoramic maps and perspective images (warping the perspective images into panoramic coordinates). To compensate for the color shift caused by the ND filter and different cameras, we follow Stumpfel *et al.* [25] to compute the color correction matrix (CCM) and align captured intensities across different settings using an X-rite classic 24 patch colorchecker. In total, our real test dataset includes 20 outdoor scenes and 40 unclipped HDR local lighting environment maps. There are 17 sunny scenes and 3 partly-cloudy scenes of roads, buildings, bridges and parks, and the sun azimuths and elevations are scattered and diverse.

4.2 Network Architecture

We model three types of lighting: ambient sky light, distant sun light, and illuminated local appearances separately in disentangled parametric spaces. To this end, SOLD-Net consists of three parts: (a) a global encoder-decoder to encode sky light and sun light separately, (b) a local encoder-renderer to encode lighting-independent local properties, *i.e.*, local contents, in the latent space and render local appearances given the global lighting conditions, and (c) a spatially-varying lighting estimator to estimate local lightings represented as disentangled components in an end-to-end manner.

Global lighting encoder-decoder. Our global lighting encoder-decoder (Figure 3 (a)) takes a global lighting panorama P_{global} as input and disentangles sky light and sun light by using two encoders to encode z_{sky} and z_{sun} in two latent spaces respectively. The sun position M_{sun} is also made explicit from the latent space of sun light for the editability of the sun position [26]. The reconstructed global lighting panorama P'_{global} is composed of reconstructed sky light P'_{sky} and reconstructed sun light P'_{sun} following Eq. (1).

Local content encoder-renderer. As shown in Figure 3 (b), the local content encoder takes a local lighting panorama P_{local} as input and extracts the lighting-independent component of local content encoded in z_{local}, which is achieved by local identity loss and cross rendering loss (in Sec. 4.3). To get the rendered local appearance P'_{app}, the local appearance renderer takes z_{local}, z_{sky}, z_{sun}, and a panoramic cosine mask M_{cos} as input to function as Eq. (2), where z_{sky} and z_{sun} are the encodings of sky light and sun light in the global lighting condition of P_{global} corresponding to the local lighting panorama P_{local}. The cosine mask M_{cos} is derived from the sun position M_{sun} indicating the shading on a hypothesized Lambertian sphere in the panoramic coordinate. Since the silhouette is a lighting-independent property, we use a silhouette decoder to estimate the silhouette

mask M'_{sil} of the local content from z_{local} as only input. The reconstruction of spatially-varying local lighting P'_{local} is then derived following Eq. (3).

Spatially-varying lighting estimator. Since we model the outdoor lighting in three disentangled latent spaces, our spatially-varying lighting estimator predicts lighting components as latent codes instead of directly estimating panoramic lighting maps. Our estimator is a single-in-multi-out network (Figure 3 (c)), which only takes a limited-FoV LDR image I of the outdoor scene as input. After extracting deep features using a shared network backbone, the sun position M'_{sun}, sky light code z'_{sky}, and sun light code z'_{sun} of global lighting are estimated in different output branches. Here we estimate the sun position as an 8×32 classification task. To estimate spatially-varying sun visibility z_{vis} caused by occlusions or weather at each pixel position, we also use a branch to predict a sun visibility mask M'_{vis} to approximately indicate whether the object on the given pixel position can be directly illuminated by the sun.For local content estimation, the stacked hourglass network [20] is used to capture the pixel-aligned features in multiple scales. Given the pixel position $l(u, v)$ on the input image, the local features at the same position in the image coordinate are extracted [22], then the local content code z'_{local} at the corresponding 3D point $L(x, y, z)$ is estimated from the extracted pixel-aligned local features. Our local lighting estimator runs in an end-to-end manner and estimates local lighting as editable disentangled lighting components.

4.3 Training

As illustrated in Figure 3, our full pipeline is trained with both supervised and self-supervised signals.

Supervised Losses. Since we are using rendered synthetic dataset, we can get the direct supervisions for many components in the pipeline. During the training of global lighting and local appearance encoding, we use the ℓ_1 reconstruction loss for P'_{sky}, P'_{app}, M'_{sil}, and the ℓ_2 reconstruction loss for P'_{sun}. Once our global and local encoders are trained, we run them on our synthetic dataset to get the disentangled codes as training targets. When training the estimator, we use the ℓ_2 reconstruction loss for M'_{vis}, z'_{sky}, z'_{sun}, and z'_{local}, and the cross-entropy loss (CE) for M'_{sun}. Please refer to the supplemental materials for more details.

Self-supervised Losses. In our formulation, local content have spatially-varying and lighting-independent properties. Therefore, the local content codes should also be lighting-independent. That is to say: 1) local lighting maps with different global lighting conditions and the same local content should have the same local content codes; 2) the local content codes encoded from the local lighting maps with the same local content should produce the same local appearance renderings with the same global lighting conditions. The constraints can be added by the local identity loss (ID) $\mathcal{L}_{\text{id}}$ and the cross rendering loss (CR) $\mathcal{L}_{\text{cr}}$. As said in Section 4.1, our synthetic dataset has paired local lighting maps $\{P^1_{\text{local}}, P^2_{\text{local}}\}$ with the same local content but different global lighting conditions, and let $\{z'^1_{\text{local}}, z'^2_{\text{local}}\}$ be the local content codes encoded by our local

encoder, then we define:

$$\mathcal{L}_{\text{id}} = ||z'^{1}_{\text{local}} - z'^{2}_{\text{local}}||_1, \tag{4}$$

and

$$\begin{aligned}\mathcal{L}_{\text{cr}} = \; & ||P^{1}_{\text{local}} \odot M_{\text{sil}} - P'^{1}_{\text{local}}(z'^{2}_{\text{local}}) \odot M_{\text{sil}}||_1 + \\ & ||P^{2}_{\text{local}} \odot M_{\text{sil}} - P'^{2}_{\text{local}}(z'^{1}_{\text{local}}) \odot M_{\text{sil}}||_1,\end{aligned} \tag{5}$$

where P'^{i}_{local} denotes rendering the local appearance map from local content codes using the global lighting condition of P^{i}_{local}. We also adopt the info loss (IF) $\mathcal{L}_{\text{info}}$ in HDSky [26] to ensure a better disentanglement of sky light and sun light.

4.4 Inference

At inference time, the inputs are a limited-FoV image and a designated pixel position. As shown in Figure 3(c), the global lighting codes of sun light and sky light, sun position and sun visibility mask are predicted. Then local content code and local sun visibility are predicted at the given pixel position. The codes are further used to decode and render global and local lighting map predictions by trained global decoder and local renderer following the formulation in Sec. 3. To estimate local lighting at multiple pixel positions, only local feature extraction, local content code estimation and local appearance rendering are needed, which means our method is of high re-usability of network predictions.

5 Experiments

In this section, we perform detailed network analysis and present qualitative and quantitative results on both synthetic data and real data, we also show relighted bunny results and virtual object insertion results to validate our methods qualitatively.

Baseline methods. To compare with spatially-varying local lighting estimation methods, we choose the latest **SOLID-Net** [28] as the baseline. To compare with global lighting estimation methods, we use **SkyNet** [11] as the baseline. To compare with fully parametric lighting models, we set two baseline $\textbf{Ours}_{\textbf{SH}}$ and $\textbf{Ours}_{\textbf{SG}}$ that use the same network architecture as our estimator to predict spatially-varying local lighting represented using 5-th order SH coefficients (108 parameters in total) and 24 SG lobes following [19] (144 parameters in total). All baselines are trained on the same synthetic dataset proposed with our method, and the implementation details can be found in supplemental materials.

Metrics. To measure the accuracy of predicted global lighting maps P'_{global}, local appearance maps P'_{app}, local lighting maps P'_{local}, and the relighted objects, we use mean absolute error (MAE) and root mean square error (RMSE) as error metrics. To measure the sun position prediction, we use angular error, azimuth error, and elevation error as error metrics.

Table 1. Quantitative evaluation of our local encoder-renderer on our enhanced synthetic dataset. 'Reconstruction' and 'Cross Rendering' denote the errors of reconstructed and cross-rendered panoramic HDR lighting maps respectively before tone mapping.

Method	Reconstruction		Cross Rendering	
	MAE	RMSE	MAE	RMSE
Ours	**0.028**	**0.075**	**0.031**	**0.079**
Ours w/o M_{cos}	0.034	0.083	0.036	0.085
Ours w/o $\mathcal{L}_{\text{cr}}$	0.029	0.083	0.048	0.101
Ours w/o $\mathcal{L}_{\text{id}}$	0.032	0.087	0.035	0.089

5.1 Ablation Study

Effectiveness of local content encoder-renderer. To validate that our local content encoder-renderer learns the lighting-independent local content instead of the lighting-dependent local appearance in z_{local}, we design an experiment of cross rendering: A pair of local lighting maps $\{P^1_{\text{local}}, P^2_{\text{local}}\}$ (same local content, different global lighting) is encoded to z^1_{local} and z^2_{local} respectively, then the local codes are swapped and the cross-rendered local appearances are obtained using the swapped local codes and the original global lighting conditions. The cross-rendered results are expected be close to ground truth local appearances.

As shown in Table 1, our local content encoder-renderer achieves a good performance[4] in cross rendering, as well as reconstruction. Without panoramic cosine mask M_{cos} (Ours w/o M_{cos}), the direction of sun light becomes unclear and the rendering performance is damaged both in reconstruction and cross rendering. Without local identity loss $\mathcal{L}_{\text{id}}$ in Eq. (4) (Ours w/o $\mathcal{L}_{\text{id}}$) or cross rendering loss $\mathcal{L}_{\text{cr}}$ in Eq. (5) (Ours w/o $\mathcal{L}_{\text{cr}}$), the cross rendering performance is heavily downgraded, indicating that the local encoder would fail to encode lighting-independent information into z_{local} without $\mathcal{L}_{\text{cr}}$ or $\mathcal{L}_{\text{id}}$.

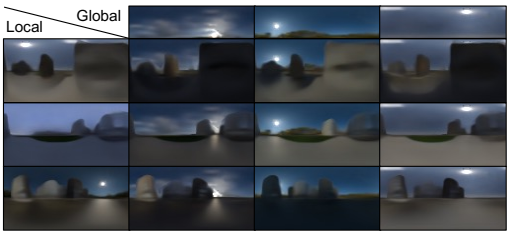


Fig. 5. Cross-rendered results of local lighting under different combinations of local contents and global lighting conditions on our enhanced synthetic dataset (shown in tone-mapped HDR). Different rows and columns indicate different local content information and global lighting conditions respectively.

Editability of disentangled lighting representation. To illustrate the editability of our disentangled lighting representation, Figure 5 shows the cross rendering results under different combinations of local contents and global lighting conditions. The first row and the first column show the reconstructed global

[4] Errors are calculated on the masked region by the (predicted) local silhouette masks.

Table 2. Quantitative evaluation of spatially-uniform global lighting estimation on our enhanced synthetic dataset. 'Panorama' denotes the errors of panoramic HDR lighting maps before tone mapping. 'Relighting' denotes the errors of rendered HDR images using predicted lighting maps.

Method	Panorama		Relighting	
	MAE	RMSE	MAE	RMSE
Ours	0.439	7.607	**0.098**	**0.119**
SkyNet [11]	0.431	8.357	0.226	0.253
SOLID-Net [28]	**0.384**	**6.360**	0.153	0.174

lighting maps and local lighting maps from our network respectively. These maps correspond to different encoded global lighting and local content information. Here we use different combinations of global lighting information and local content information to cross-render corresponding local lighting maps. In each row, the same local content code z_{local} is shared and the layouts of local content are basically the same, which further validates that the lighting-independent local content information is indeed encoded into z_{local}. Given global lighting codes z_{sky} and z_{sun} in each column, local appearances are correctly rendered, such as blocking of the sun light by the buildings and reflection of the sun light on the ground when the sun is at a low elevation angle. The results show our method's editability of the local lighting. Our disentangled lighting representation also allows the editing of the global lighting since the sun light, sky light and sun position are represented in separate spaces (please refer to supplemental material).

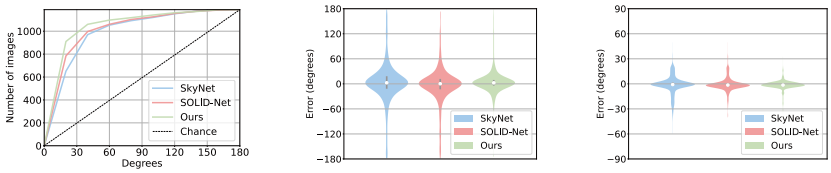
Importance of sun visibility mask. In our method, a predicted sun visibility mask M'_{vis} is used to indicate the effects of the weather and the spatially-varying occlusion of the sun, since whether an object is directly illuminated by the distant sun light will affect its appearance significantly. Such a design is verified by removing the prediction branch of M_{vis} in our lighting estimator and fix $z_{\text{vis}} = 1$ (Ours w/o M_{vis}). As we can see from Table 3, without using the information from the predicted sun visibility mask M'_{vis} to constrain the predicted distant sun light component in local lighting, the accuracy of both reconstructed environment maps and relighting results will be degraded.

5.2 Experimental Results

Spatially-uniform global lighting estimation. We compare global lighting estimation performance of our method (Ours), SkyNet [11], and SOLID-Net [28].

To evaluate the sun position estimation, we compute sun angular error, azimuth error, and elevation error on our test set, as shown in Figure 6. The sun position is calculated as the centroid of the largest connected component of the global lighting panorama above a threshold (98%) [28]. We can see that by treating sun position estimation as a disentangled task, our method achieves favorable improvement over other baseline methods in sun position estimation.

For quantitative evaluation, we calculate the errors on estimated global lighting of the sky-dome represented in panoramic maps (only the upper half of the predicted global environment maps by SOLID-Net [28] are used for calculation).



(a) Sun angular error (b) Azimuth error (c) Elevation error

Fig. 6. Quantitative evaluation of sun position estimation on our enhanced synthetic dataset. (a) Cumulative curves of the sun angular error of baseline methods and ours. (b) Azimuth errors and (c) elevation errors of the sun position estimation are displayed as 'violin plots', where the envelope of each bin represents the percentile, the gray line represents the percentile of 25% to 75%, and the white point represents the median.

Table 3. Quantitative evaluation of spatially-varying local lighting estimation on our enhanced synthetic dataset. 'Panorama' denotes the errors of panoramic HDR lighting maps before tone mapping. 'Relighting' denotes the errors of rendered HDR images using predicted lighting maps.

Method	Panorama		Relighting	
	MAE	RMSE	MAE	RMSE
Ours	**0.128**	2.394	**0.075**	0.145
Ours w/o M_{vis}	0.140	2.814	0.081	0.159
Ours$_{\text{SH}}$	0.190	**1.943**	0.081	**0.139**
Ours$_{\text{SG}}$	0.170	2.785	0.093	0.179
SOLID-Net [28]	0.308	3.384	0.186	0.337

The panoramas are rotated before evaluation to move the sun to its center [11]. Furthermore, we evaluate the rendering errors[5] on the relighting of a glossy Stanford Bunny [4] using predicted global lighting maps. For panoramic errors in global lighting map estimation, we can see from Table 2 that our method is on par with SkyNet [11] and SOLID-Net [28]. However, our method performs significantly better on relighting results. This is because metrics on outdoor lighting maps are sensitive to even a slight misalignment of high-intensity pixels, while metrics on relighting results are more robust and close to human perception.

From the practical point of view, the relighting performance is of more concern. As shown in Figure 7, our method better preserves the high-frequency information in sun light and our relighting results look more realistic than those of SkyNet [11] and SOLID-Net [28], which predict less high-frequency lighting component resulting in lower errors of panoramic lighting maps.

Spatially-varying local lighting estimation. We compare local lighting estimation performance of our method (Ours), Ours w/o M_{vis}, Ours$_{\text{SG}}$, Ours$_{\text{SH}}$, and SOLID-Net [28].

A quantitative evaluation of estimated panoramic local lighting maps and the relighting results by the estimated local lighting maps on our test set is shown in Table 3. Our method significantly outperforms the non-parametric method SOLID-Net [28] in both reconstruction and relighting results. The fully

[5] We calculate rendering errors on pixels belonging to the bunny body.

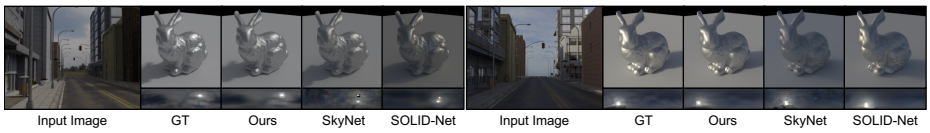


Fig. 7. Relighting results with global lighting maps on our enhanced synthetic dataset.

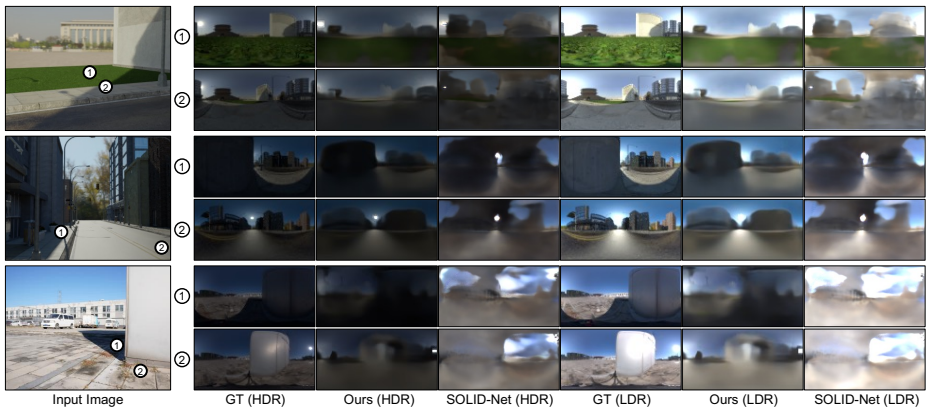


Fig. 8. Qualitative comparison of local lighting estimation results. Column 1 shows the input image and two selected pixel positions. Column 2 and column 5 show the ground truth local lighting environment map in HDR and LDR respectively. Columns 3-4 and columns 6-7 show the estimated local lighting by our method and SOLID-Net [28] in HDR and LDR respectively. The first two rows are from our enhanced synthetic dataset and the last row is from our captured real data.

parametric methods Ours_{SH} and Ours_{SG} generally perform well, which shows the effectiveness of our local feature extraction network allowing spatially-varying prediction in both SH and SG models. Overall, our method achieves state-of-the-art performance with more flexible editability than other methods.

As shown in the qualitative comparison of local lighting estimation (Figure 8), our method can recover the major layout of local lighting maps, render local appearances given estimated global lighting conditions, and correctly predict sun visibility, while SOLID-Net [28] suffers from the accumulated errors of the scene geometry estimation and the lack of explicit constraint of sun visibility. Our method does not rely on explicitly estimating accurate geometry and can recover a cleaner environment map with fewer high-frequency artifacts.

We also show relighted bunny results in Figure 9 for further qualitative comparison of estimated local lighting on our captured real data, which is more challenging due to the dataset shift. We can see that though Ours_{SH} performs well on pixel average, the relighting results are visually unrealistic due to the lack of hard shadows. Ours_{SG} is not as robust as Ours_{SH} on real data. SOLID-Net [28] is misled by inaccurately estimating the complex geometry and shadow clues. Our method performs better visually on both dark and bright regions.

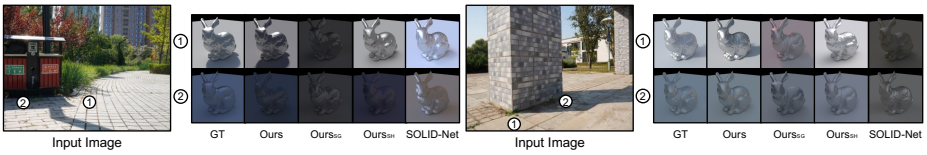


Fig. 9. Qualitative comparison of relighting results on our captured real data.

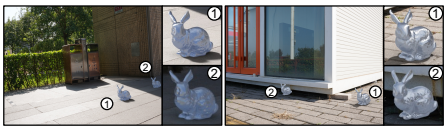


Fig. 10. Examples of virtual object insertion on our captured real data.

Fig. 11. Examples of local lighting estimation on in-the-wild data.

Virtual object insertion. A useful application of spatially-varying local lighting estimation is to insert a virtual object into the scene. As shown in Figure 10, the inserted objects can be realistically relit at the positions of insertions.
Generalization ability. Although our pipeline is trained on fully synthetic dataset of urban scenes, the spatially-varying properties of lighting in natural scenes can also be captured. Besides, thanks to the synthetic dataset enhancement, our pipeline can work for non-sunny weather, as shown in Figure 11.

6 Conclusion

In this paper, we propose a novel parametric lighting model with disentangled spaces and formulate lighting estimation on outdoor scenes as an end-to-end learning problem. The major benefit of the proposed method is that it learns a flexible lighting representation that is user-friendly to manipulate. In addition, neither explicit intrinsic estimations nor time-consuming optimization procedures for traditional parametric lighting models are not needed. From extensive experiments and qualitative demonstrations, the effectiveness of the proposed method is verified and the proposed method achieves state-of-the-art performance compared with previous work on both synthetic and real data.
Limitations and future work. While our method performs well in estimating at bright regions, it may fail to handle the sharp lighting change near the boundary pixels. Exploring more spatially-coherent local lighting estimation [24] is also an interesting direction for our future work. Besides, our lighting editing only faithfully modifies disentangled components, and automatic harmonization to make the edited lighting more natural will benefit user interaction in the future. A large-scale dataset for spatially-varying lighting in more diverse outdoor scenes will be helpful to boost the training of learning-based methods.
Acknowledgements. This work is supported by National Natural Science Foundation of China under Grant No. 62136001, 61872012.

References

1. ambientCG, public domain materials for physically based rendering. [licensed under CC0 1.0 universal], https://ambientcg.com 6
2. Blender SceneCity., https://www.cgchan.com/store/scenecity 2, 5, 6
3. Poly Haven, The Public 3D Asset Library. [licensed under CC0 1.0 universal], https://polyhaven.com 6
4. The Stanford 3D Scanning Repository. http://graphics.stanford.edu/data/3Dscanrep/ 12
5. Barron, J.T., Malik, J.: Intrinsic scene properties from a single RGB-D image. In: Proc. of Computer Vision and Pattern Recognition (2013) 4
6. Burley, B.: Physically-based shading at Disney. In: SIGGRAPH course: Practical Physically Based Shading in Film and Game Production (2012) 6
7. Cheng, D., Shi, J., Chen, Y., Deng, X., Zhang, X.: Learning scene illumination by pairwise photos from rear and front mobile cameras. Computer Graphics Forum **37**(7), 213–221 (2018) 3
8. Debevec, P.: Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In: Proc. of ACM SIGGRAPH (1998) 3
9. Gardner, M.A., Hold-Geoffroy, Y., Sunkavalli, K., Gagné, C., Lalonde, J.F.: Deep parametric indoor lighting estimation. In: Proc. of International Conference on Computer Vision (2019) 4
10. Garon, M., Sunkavalli, K., Hadap, S., Carr, N., Lalonde, J.F.: Fast spatially-varying indoor lighting estimation. In: Proc. of Computer Vision and Pattern Recognition (2019) 4
11. Hold-Geoffroy, Y., Athawale, A., Lalonde, J.F.: Deep sky modeling for single image outdoor lighting estimation. In: Proc. of Computer Vision and Pattern Recognition (2019) 1, 2, 3, 9, 11, 12
12. Hold-Geoffroy, Y., Sunkavalli, K., Hadap, S., Gambaretto, E., Lalonde, J.F.: Deep outdoor illumination estimation. In: Proc. of Computer Vision and Pattern Recognition (2017) 3
13. Hosek, L., Wilkie, A.: An analytic model for full spectral sky-dome radiance. ACM Trans. on Graphics (TOG) **31**(4) (2012) 1, 3
14. Hosek, L., Wilkie, A.: Adding a solar-radiance function to the Hošek-Wilkie skylight model. IEEE Computer Graphics and Applications **33**(3), 44–52 (2013) 1, 3
15. Lalonde, J.F., Efros, A.A., Narasimhan, S.G.: Estimating the natural illumination conditions from a single outdoor image. International Journal of Computer Vision **98**(2), 123–145 (2012) 3
16. Lalonde, J.F., Asselin, L.P., Becirovski, J., Hold-Geoffroy, Y., Garon, M., Gardner, M.A., Zhang, J.: The laval HDR sky database. [free license for academic or government-sponsored researchers] (2016), http://sky.hdrdb.com 3
17. Lalonde, J.F., Matthews, I.: Lighting estimation in outdoor image collections. In: Proc. of International Conference on 3D Vision (2014) 1, 3
18. LeGendre, C., Ma, W.C., Fyffe, G., Flynn, J., Charbonnel, L., Busch, J., Debevec, P.: DeepLight: Learning illumination for unconstrained mobile mixed reality. In: Proc. of Computer Vision and Pattern Recognition (2019) 3
19. Li, Z., Shafiei, M., Ramamoorthi, R., Sunkavalli, K., Chandraker, M.: Inverse rendering for complex indoor scenes: Shape, spatially-varying lighting and SVBRDF from a single image. In: Proc. of Computer Vision and Pattern Recognition (2020) 4, 5, 9

20. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: Proc. of European Conference on Computer Vision (2016) 8
21. Ramamoorthi, R., Hanrahan, P.: An efficient representation for irradiance environment maps. In: Proc. of ACM SIGGRAPH (2001) 5
22. Saito, S., Huang, Z., Natsume, R., Morishima, S., Kanazawa, A., Li, H.: PIFu: Pixel-aligned implicit function for high-resolution clothed human digitization. In: Proc. of International Conference on Computer Vision (2019) 8
23. Song, S., Funkhouser, T.: Neural illumination: Lighting prediction for indoor environments. In: Proc. of Computer Vision and Pattern Recognition (2019) 4
24. Srinivasan, P.P., Mildenhall, B., Tancik, M., Barron, J.T., Tucker, R., Snavely, N.: Lighthouse: Predicting lighting volumes for spatially-coherent illumination. In: Proc. of Computer Vision and Pattern Recognition (2020) 4, 14
25. Stumpfel, J., Jones, A., Wenger, A., Tchou, C., Hawkins, T., Debevec, P.: Direct HDR capture of the sun and sky. In: Proc. of ACM SIGGRAPH (2004) 3, 7
26. Yu, P., Guo, J., Huang, F., Zhou, C., Che, H., Ling, X., Guo, Y.: Hierarchical disentangled representation learning for outdoor illumination estimation and editing. In: Proc. of International Conference on Computer Vision (2021) 1, 3, 7, 9
27. Zhang, J., Sunkavalli, K., Hold-Geoffroy, Y., Hadap, S., Eisenmann, J., Lalonde, J.F.: All-weather deep outdoor lighting estimation. In: Proc. of Computer Vision and Pattern Recognition (2019) 3
28. Zhu, Y., Zhang, Y., Li, S., Shi, B.: Spatially-varying outdoor lighting estimation from intrinsics. In: Proc. of Computer Vision and Pattern Recognition (2021) 2, 4, 5, 6, 9, 11, 12, 13

Estimating Spatially-Varying Lighting in Urban Scenes with Disentangled Representation (Supplemental Material)

Jiajun Tang[1], Yongjie Zhu[2], Haoyu Wang[1], Jun Hoong Chan[1], Si Li[2], and Boxin Shi[1,3] ✉

[1] NERCVT, School of Computer Science, Peking University
[2] School of Artificial Intelligence, Beijing University of Posts and Telecommunications
[3] Peng Cheng Laboratory

In this supplementary material, we provide more details about our data collection, implementation details and network architectures. We also show additional results on our enhanced synthetic dataset, captured real data and collected in-the-wild data.

7 Appendix

7.1 Data Collection Details

For synthetic data generation, we collect 30 PBR materials from ambientCG [1] in 4 categories: bricks, concrete, pavement, and grass field; we then mark the 3D city model with the category labels and randomly replace the materials following the category labels in Blender [2][6]. After material enhancement, we follow the camera sampling and rendering process proposed in SOLID-Net [13]. Specifically, the view point (x, y, z) of the local lighting is 10 cm away from the surface at the designated pixel position (u, v) along its normal.

When capturing the real data of the outdoor spatially-varying lighting, we use a level to ensure we capture the correct sun elevation angle relative to the skyline and we use a compass to ensure we capture the correct sun azimuth angle in the panoramas relative to the observation direction of the limited-FoV images. Due to the long capture time and the moving clouds, we discard the data if the lighting environment changes largely in its capture process, which makes our data capture very time-consuming, especially in (partly-)cloudy weathers.

7.2 Implementation Details

In our implementation, we use 128 parameters in total to model spatially-varying lighting. Among them, $z_{\text{sky}} \in \mathbf{R}^{16}$, $z_{\text{sun}} \in \mathbf{R}^{45}$, M_{sun} is determined by 2 parameters, $z_{\text{local}} \in \mathbf{R}^{64}$, and we treat the spatially-varying sun visibility z_{vis} as another parameter. The image resolutions are 320×240 for limited-FoV images I and 128×64 for panoramic maps (128×32 for panoramic maps of the sky-dome).

[6] We use the PBR materials in Blender following the document at https://docs.ambientcg.com/books/using-the-assets/page/pbr-materials-in-blender

Our full pipeline is implemented in PyTorch [10] and trained step-wise. We first train our global lighting encoder-decoder on Laval Sky dataset [6] for 30 epochs and finetune on the mixture of Laval Sky dataset [6] and our global lighting data for 5 epochs. Then we extract the global lighting codes z_{sky} and z_{sun} of our global lighting data and train our local content encoder-renderer using our local-global paired data for 20 epochs. At last, the local lighting data are encoded into z_{local} and we train our spatially-varying lighting estimator using these code labels for 20 epochs. When training the global lighting encoder-decoder and the local content encoder-renderer, we apply the radiometric distortion [3] and random rotation at the probability of 0.5 and 0.6 respectively, and we use the ground truth sun position as input, leaving the sun position estimation as a disentangled task in the estimator. The batch size is set to 16 in all of our experiments. The sun position map M_{sun} has the resolution of 128×32 and only the 8×8 patch at the (predicted) sun position is set as 1 otherwise 0. We use Adam optimizer [5] in all of our experiments, the initial learning rates are set to 4×10^{-3} and are halved every 5 epochs.

Since outdoor HDR images have extremely bright intensities in sun pixels, for a more stable training using CNN, all HDR images are tone-mapped before being fed into the networks [4]:

$$T = \frac{\log(1 + \mu H)}{\log(1 + \mu)}, \tag{6}$$

where linear HDR images H are mapped into log-compressed images T, and we empirically set $\mu = 16$.

7.3 Sun Visibility Mask

In our proposed methos, we unify the occlusion and weather factors into one sun visibility component. The sun visibility mask M_{vis} used in our method has the same resolution as the limited-FoV image I and indicates whether a pixel position is directly illuminated by the sun light, which is not identical to a shadow mask in non-sunny weathers. However it's hard to get the accurate mask strictly following the aforementioned definition. To get the approximate mask for training supervision, we use the Lambertian diffuse material to override the original materials in the 3D city model and render limited-FoV images using the same cameras with the shadow visibility set on and off in the Blender [2] Cycles engine as I_{ws} and I_{wos}, then we calculate the intensity of the difference map $D = \text{gray}(I_{\text{wos}} - I_{\text{ws}})$. The higher pixel value in D, the more likely the pixel is in the cast shadow or attached shadow and is not directly illuminated by the sun light. We use the combination of a fixed threshold of the minimum difference $t_{\text{min}} = 0.002$ and an adaptive threshold t_{otsu} from the Otsu's method [9] to convert the different map D into the approximate binary label of sun visibility mask M_{vis}. For cloudy scenes, we set all pixels as non-visible (0). Since the mask is an approximation and to make our prediction robust to noisy pixels, we use the predicted mask M'_{vis} in a conservative manner: We constrain the sun light

visibility in the estimated local lighting only when all pixel values in the 7×7 patch of the pixel position are smaller than 0.2, otherwise we do nothing.

7.4 Training Losses

Our full pipeline is trained with the following losses:

$$\mathcal{L} = \mathcal{L}_G + \mathcal{L}_L + \mathcal{L}_E, \tag{7}$$

where $\mathcal{L}_G$ is the loss term for our global lighting encoder-decoder, $\mathcal{L}_L$ is the loss term for our local content encoder-renderer, and $\mathcal{L}_E$ is the loss term for our spatially-varying lighting estimator.

For global lighting encoder-decoder, $\mathcal{L}_G$ is defined as:

$$\mathcal{L}_G = \mathcal{L}_{\text{sky}} + \mathcal{L}_{\text{sun}} + \mathcal{L}_{\text{info}}, \tag{8}$$

where $\mathcal{L}_{\text{sky}}$ is the L_1 reconstruction loss for sky light:

$$\mathcal{L}_{\text{sky}} = ||P_{\text{sky}} \odot (1 - M_{\text{sun}}) - P'_{\text{sky}} \odot (1 - M_{\text{sun}})||_1, \tag{9}$$

$\mathcal{L}_{\text{sun}}$ is the L_2 reconstruction loss for sun light:

$$\mathcal{L}_{\text{sun}} = ||P_{\text{sun}} \odot M_{\text{sun}} - P'_{\text{sun}}||_2, \tag{10}$$

and $\mathcal{L}_{\text{info}}$ is the same as that in HDSky [11].

For local content encoder-renderer, $\mathcal{L}_L$ is defined as:

$$\mathcal{L}_L = \mathcal{L}_{\text{app}} + \mathcal{L}_{\text{sil}} + \mathcal{L}_{\text{id}} + \mathcal{L}_{\text{cr}}, \tag{11}$$

where $\mathcal{L}_{\text{app}}$ is the L_1 reconstruction loss for local appearances:

$$\mathcal{L}_{\text{app}} = ||P_{\text{local}} \odot M_{\text{sil}} - P'_{\text{app}} \odot M_{\text{sil}}||_1, \tag{12}$$

and $\mathcal{L}_{\text{sil}}$ is the L_1 reconstruction loss for M_{sil}. $\mathcal{L}_{\text{id}}$ and $\mathcal{L}_{\text{cr}}$ are self-supervised losses introduced in Sec. 4.3.

For spatially-varying lighting estimator, $\mathcal{L}_E$ is defined as:

$$\mathcal{L}_E = \mathcal{L}_{\text{vis}} + \mathcal{L}_{\text{pos}} + \mathcal{L}_{\text{code}} + \mathcal{L}_{\text{pano}}, \tag{13}$$

where $\mathcal{L}_{\text{vis}}$ is an L_2 loss for M_{vis}, $\mathcal{L}_{\text{pos}}$ is the cross-entropy loss for flattened M_{sun}, $\mathcal{L}_{\text{code}}$ is an L_2 loss for z_{sky}, z_{sun}, and z_{local}. We also impose $\mathcal{L}_{\text{pano}}$, an L_1 loss on decoded/rendered panoramic maps P'_{sky}, P'_{sun}, P'_{sil}, and P'_{app}.

7.5 Implementation of Baseline Methods

SOLID-Net [13]. We train SOLID-Net [13] on our enhanced dataset using their original code implementation. Following their paper, we render and compute normal maps, depth maps, (diffuse) shading maps, (diffuse) shadow maps and albedo maps as learning targets to train their I-Net. Following their geometry

projection, we prepare incomplete panoramas projected from the limited-FoV images at different local positions as the input of their P-Net. Their I-Net and P-Net are trained separately for 30 epochs and 20 epochs respectively.

SkyNet [3]. We implement SkyNet [3] in PyTorch [10] following the network structure shown in their paper. SkyNet [3] is designed to separately predict a sun azimuth angle and a normalized panoramic environment map where the sun is azimuth-centered. To fit their data requirement, we compute the sun azimuth of our data and compute the normalized panoramic environment maps as the training targets of SkyNet [3]. Their sky autoencoder and image encoders are trained separately for 30 epochs and 20 epochs respectively.

Ours$_{\mathbf{SH}}$. We use the same network backbone for Ours$_{\text{SH}}$ as our proposed method (see Sec. 7.7) to predict 5-th order SH coefficients as representation for local lighting. The ground truth SH coefficients can be directly computed from the orthogonality of SH basis. The model is trained for 30 epochs.

Ours$_{\mathbf{SG}}$. We use the same network backbone for Ours$_{\text{SG}}$ as our proposed method (see Sec. 7.7) to predict 24 SG lobes as representation for local lighting. The corresponding parameters of SG lobes may not be unique given an environment map, which poses a problem to use SG parameters as learning targets. Here we follow [7] to reparameterize SG parameters to constrain each lobe in the certain range of the sphere (we roughly divide the sphere into 3×8 regions) and optimize the ground truth SG parameters by LBFGS method [8]. Note that this process is very time-consuming[7], meaning that using optimized SG lobes to represent outdoor lighting might not be an efficient choice. The model is trained for 30 epochs.

7.6 Additional Results

Qualitative results. We show the quantitative results of sun position estimation on our captured real data in Table 4[8]. We can see from the figures that though the real data are more challenging than synthetic dataset, our method still achieves a better performance than baseline methods. It's worth noting that SOLID-Net [13] trained on synthetic dataset *before enhancement* (SOLID-Net w/o en) shows an significant drop compared with the same model trained on synthetic dataset *after enhancement* (SOLID-Net), indicating the enhanced material diversity is helpful for real-world performance. We also show the quantitative evaluation of local lighting estimation and relighting performance on our captured real data in Table 5, and the trends are similar as on our enhanced synthetic dataset, indicating our method can consistently produce more realistic relighting results.

Synthetic dataset enhancement. To further confirm the effectiveness of our synthetic dataset enhancement, we test the SOLID-Net [13] trained on synthetic

[7] https://github.com/lzqsd/InverseRenderingOfIndoorScene#differences-from-the-original-paper

[8] Note that Ours$_{\text{SH}}$ and Ours$_{\text{SG}}$ is not designed for global sun position estimation, we use the maximum points of the predicted local lighting maps given the pixel positions in the bright regions as the predicted sun positions only as a reference here.

dataset without and with enhancement on the same real-captured data, and the qualitative results are shown in Figure 12. As we can seen, the SOLID-Net [13] with enhanced dataset (blue) can recover clearer scene layouts and colors in ground truth (green) than without enhancement (red) with the presence of irregular grasses and leafs, which would lead to noisy intrinsic estimation results.

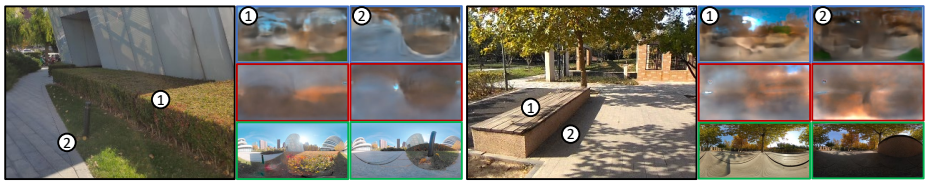


Fig. 12. Local lighting estimation results on real data of SOLID-Net [13] trained on synthetic dataset without enhancement (the first row, blue box) and with enhancement (the second row, red box), compared with the ground truth local lighting (the third row, green box). Panoramas are shown in tone-mapped HDR.

Controlled scene test. To better understand the bounds of our proposed method, we conduct a controlled scene test in Figure 13. As we can see, our method works well as expected in the sunny (hard shadow) urban scenes and can generalize to (partly-)cloudy (soft shadow) weathers (Figure 13 left and middle) and different proximity to the buildings (Figure 13 middle and right). The two known common types of failure cases (Figure 13 left) happen around complex local shapes (blue point) and highlight glass reflections (orange point).

Fig. 13. Local lighting estimation and relighting results of controlled scene test. The red (green) points in three images correspond to the same spot which is sun-(in)visible. The organge and blue points show two typical types of failure cases. Colors of boxes indicate the correspondence with local pixel points marked in the same color. Panoramas and relighting results are shown in tone-mapped HDR.

Editability. To further illustrate the editability of our disentangled representation of global and local lighting, we show more qualitative examples. For global lighting editing, we show examples of sun position editing in Figure 14, sun information editing in Figure 15, and sky information editing in Figure 16. For local lighting editing, we show more results in Figure 17. We can see from the results that by

Table 4. Quantitative evaluation of sun position estimation on our captured real data. We report the average angular error $\mathcal{E}_{\text{ang}}$, azimuth error $\mathcal{E}_{\text{az}}$, and elevation error $\mathcal{E}_{\text{el}}$ in degrees. The results of Ours_{SH} and Ours_{SG} are calculated from the predicted local lighting maps for reference purpose. Lower is better.

Method	$\mathcal{E}_{\text{ang}}$	$\mathcal{E}_{\text{az}}$	$\mathcal{E}_{\text{el}}$
Ours	**21.15**	**20.76**	**8.44**
SkyNet [3]	32.83	32.81	14.20
SOLID-Net [13]	30.82	33.92	11.95
SOLID-Net [13] w/o en	49.07	61.20	**8.03**
$\textbf{Ours}_{\textbf{SH}}$	48.91	64.55	21.96
$\textbf{Ours}_{\textbf{SG}}$	60.87	68.84	24.31

Table 5. Quantitative evaluation of spatially-varying local lighting estimation on our captured real data. 'Panorama' stands for the errors of panoramic HDR lighting maps before tone mapping. 'Relighting' stands for the errors of rendered HDR images using predicted lighting maps. Lower is better.

Method	Panorama		Relighting	
	MAE	RMSE	MAE	RMSE
Ours	0.240	4.872	**0.259**	**0.437**
Ours w/o M_{vis}	0.274	6.496	0.299	0.552
$\textbf{Ours}_{\textbf{SH}}$	0.244	**2.367**	0.269	0.469
$\textbf{Ours}_{\textbf{SG}}$	**0.179**	2.748	0.318	0.560
SOLID-Net [13]	0.390	4.206	0.310	0.601

predicting disentangled global and local representations according to our problem formulation, the predicted global and local lighting remain flexible editability.

Quantitative results. More quanlitative results of global lighting estimation and local lighting estimation are shown in Figure 18 and Figure 19 respectively. In each figure, we show the estimated lighting maps, relighting results using the (predicted) lighting maps and the quantitative error metrics of the lighting maps and relighting results. Our method generally performs better qualitatively than baseline methods while may not be the best in the error metrics, showing the inconsistency of the quantitative metrics and visual perception.

In-the-wild performance. To test the generalization ability of the compared methods, we show the local relighting results of in-the-wild data in Figure 20 and Figure 21. The in-the-wild data are collected from the Google Street View dataset [12] and Internet photos, which cover diverse scenes and lighting conditions. Although the data distribution is far different from our synthetic training data, our proposed method can capture many spatially-varying properties of outdoor local lighting and give reasonable relighting results.

7.7 Detailed Network Archetectures

We show the detailed network archtectures of the global lighting encoder-decoder, local content encoder-renderer, spatially-varying lighting estimator and our baseline methods Ours_{SH} and Ours_{SG} from Table 6 to Table 9, with the structures and default settings of common blocks shown in Figure 22.

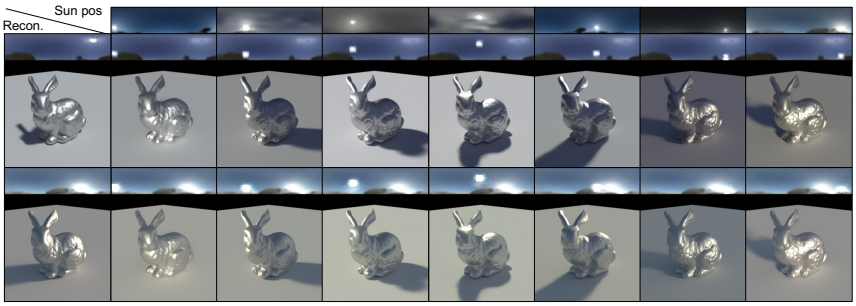


Fig. 14. Global lighting editing by changing the sun positions (in each row) and the relighting results (shown in tone-mapped HDR).

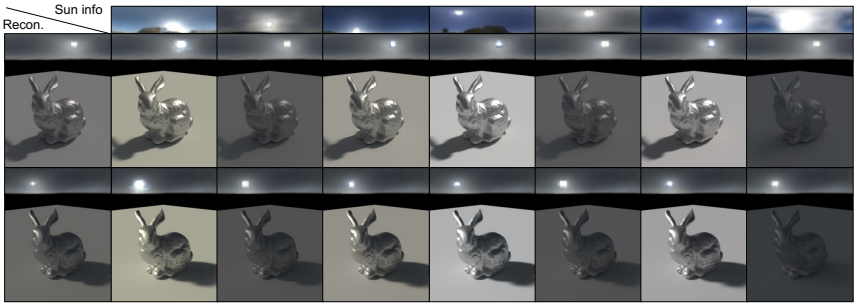


Fig. 15. Global lighting editing by changing the sun info (in each row), and the relighting results (shown in tone-mapped HDR).

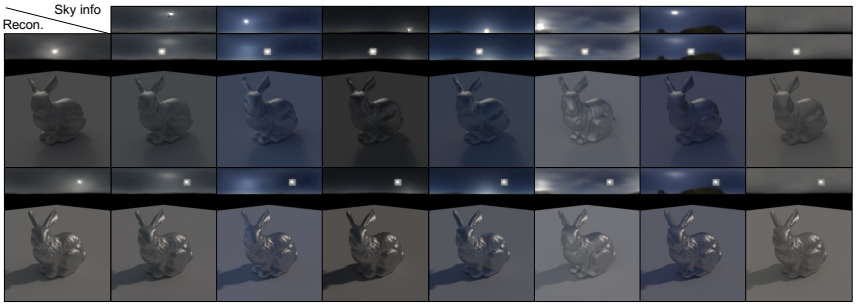


Fig. 16. Global lighting editing by changing the sky info (in each row), and the relighting results (shown in tone-mapped HDR).

Fig. 17. Local lighting editing by changing global lighting conditions (in each row) and local contents (in each column), and the relighting results (shown in tone-mapped HDR).

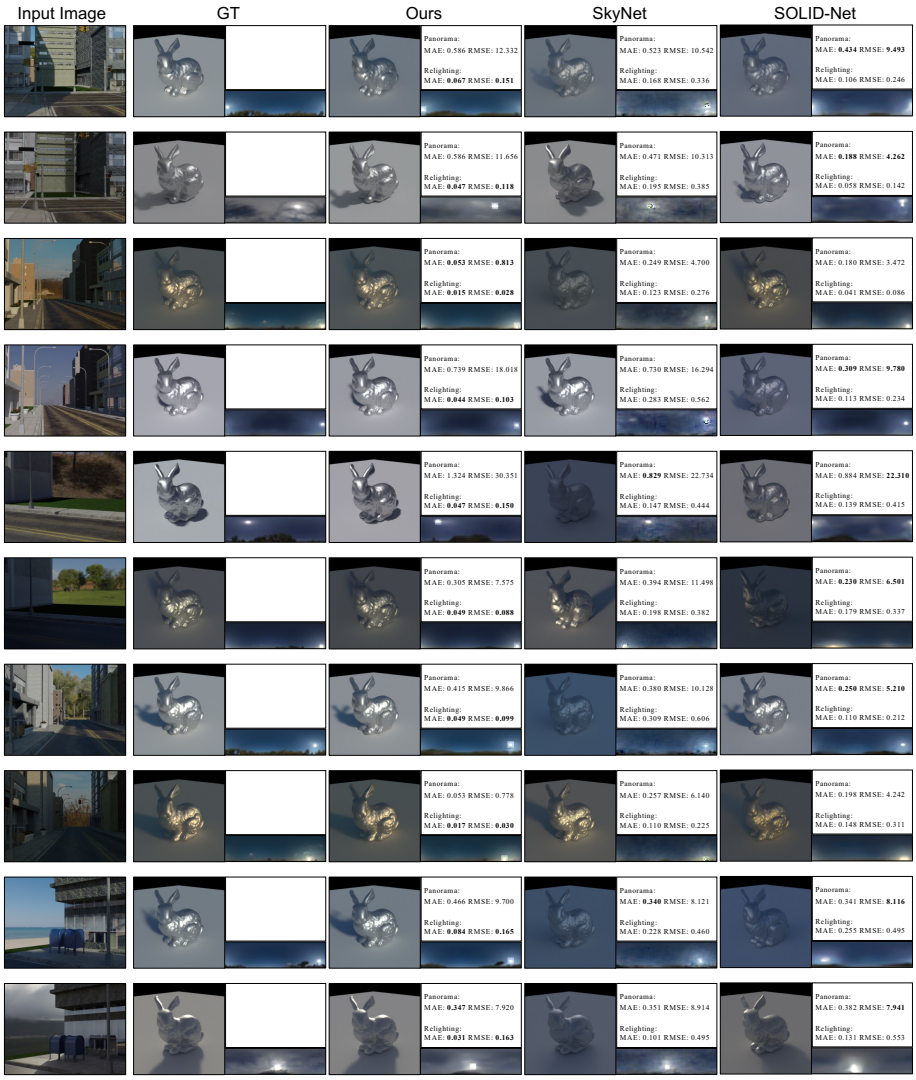


Fig. 18. Global lighting estimation and relighting results on our enhanced synthetic dataset (shown in tone-mapped HDR). Errors of instances are shown in text boxes.

Fig. 19. Local lighting estimation and relighting results (shown in tone-mapped HDR). The first three examples are from our enhanced synthetic dataset, and the last five examples are from our captured real data. Errors of instances are shown below each image (from left to right: MAE of panoramic maps, RMSE of panoramic maps, MAE of relighting results and RMSE of relighting results).

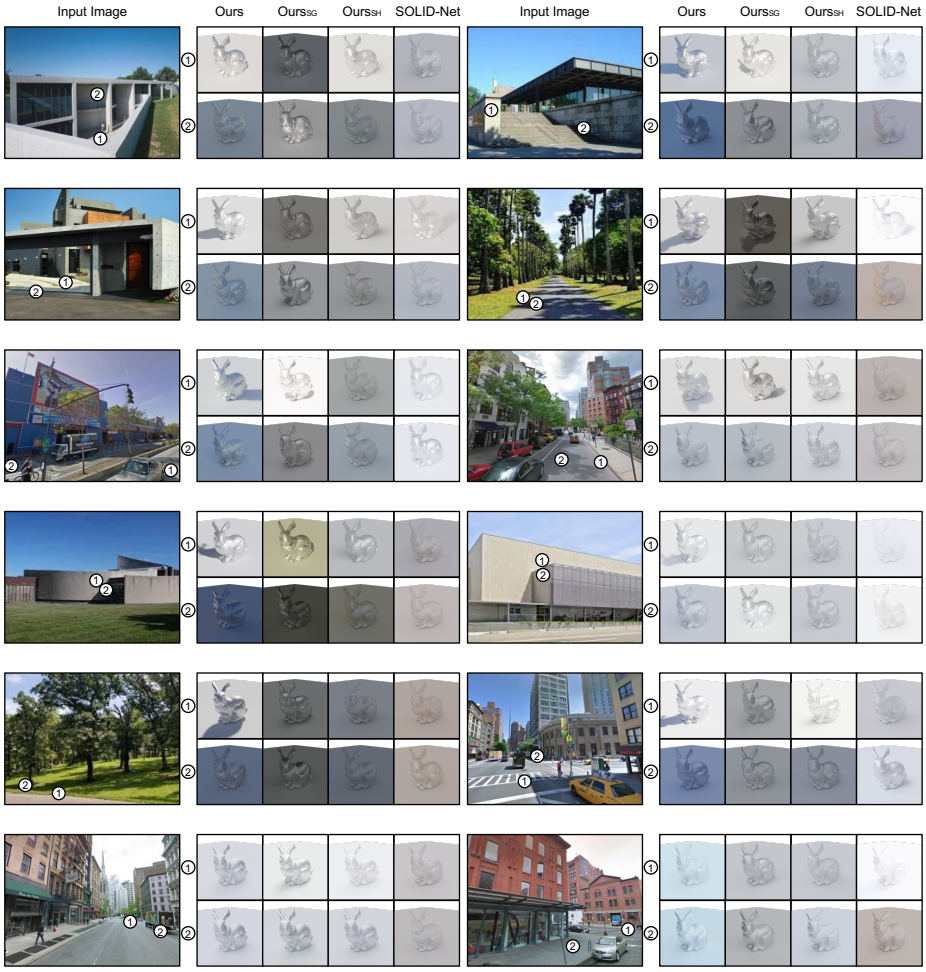


Fig. 20. More examples of local lighting estimation on in-the-wild data.

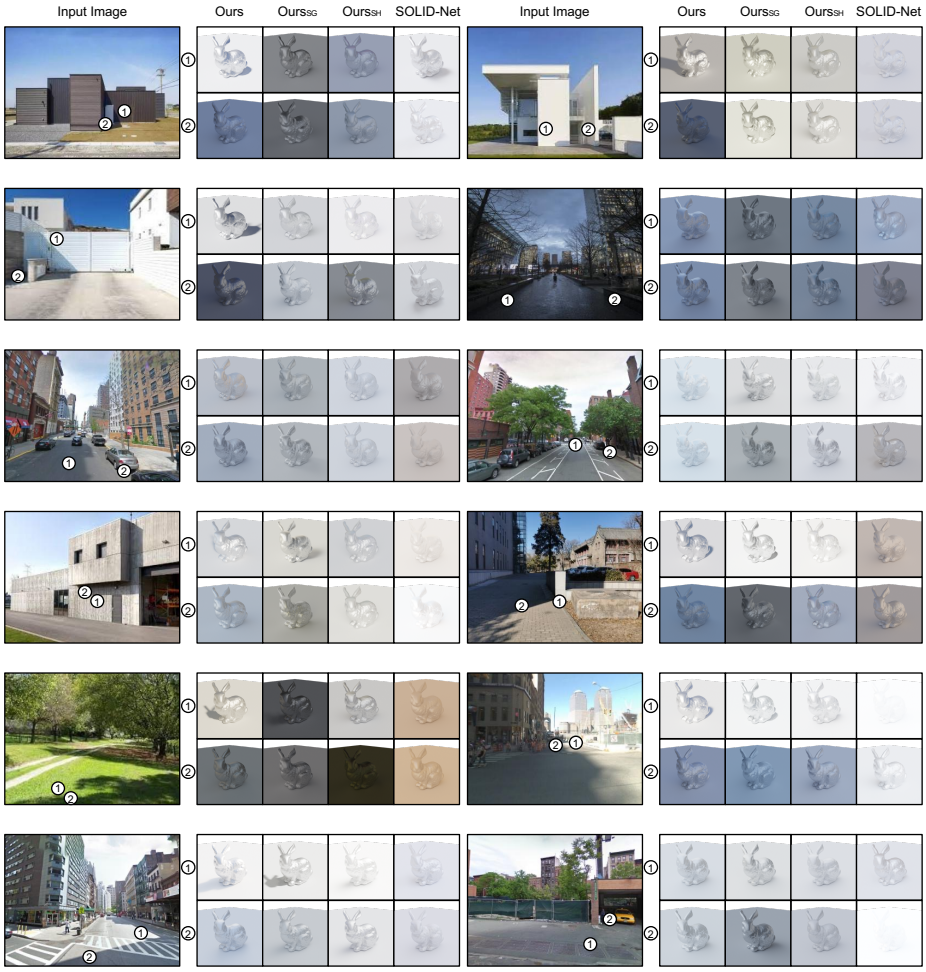


Fig. 21. More examples of local lighting estimation on in-the-wild data.

Table 6. Network architectures of our global lighting encoder-decoder.

Global Lighting Encoder-Decoder			
Name	Layer Description	Input	Output Dim.
P_{global}	Global lighting maps	–	(3, 64, 128)
z_{sky}	Latent code for sky light	P_{global}	(16)
z_{sun}	Latent code for sun light	P_{global}	(45)
M_{sun}	Mask for sun position	P_{global}	(1, 64, 128)
Encoders for z_{sky} ($c_{\text{out}} = 16$) and z_{sun} ($c_{\text{out}} = 45$)			
Conv1_1	Conv., k=5, s=1	P_{global}	(32, 32, 128)
Conv1_2	Res. I, k=3, s=1	Conv1_1	(32, 32, 128)
Conv1_3	Conv., k=3, s=2	Conv1_2	(64, 16, 64)
Conv2_1	Conv., k=3, s=1	Conv1_3	(64, 16, 64)
Conv2_2	Res. I, k=3, s=1	Conv2_1	(64, 16, 64)
Conv2_3	Conv., k=3, s=2	Conv2_2	(128, 8, 32)
Conv3_1	Conv., k=3, s=1	Conv2_3	(128, 8, 32)
Conv3_2	Res. I, k=3, s=1	Conv3_1	(128, 8, 32)
Conv3_3	Conv., k=3, s=2	Conv3_2	(128, 4, 16)
Conv4_1	Conv., k=3, s=1	Conv3_3	(128, 4, 16)
Conv4_2	Res. I, k=3, s=1, no norm	Conv4_1	(128, 4, 16)
Conv4_3	Conv., k=3, s=2, no norm	Conv4_2	(64, 2, 8)
Conv5	Conv., k=3, s=1, no norm, no activ	Conv4_3	(c_{out}, 2, 8)
Pool	Global Avg. Pool.	Conv5	(c_{out})
Decoder for P_{sky}			
Fc	Linear 16×512, and reshape	z_{sky}	(8, 4, 16)
Conv1_1	Conv., k=3, s=1	Fc	(64, 4, 16)
Conv1_2	2×Res. I, k=3, s=1	Conv1_1	(64, 4, 16)
Up1	Deconv., k=3, s=1	Conv1_2	(64, 8, 32)
Conv2_1	Conv., k=3, s=1	Up1	(128, 8, 32)
Conv2_2	2×Res. I, k=3, s=1	Conv2_1	(128, 8, 32)
Up2	Deconv., k=3, s=1	Conv1_2	(128, 16, 64)
Conv3_1	Conv., k=3, s=1	Up2	(64, 16, 64)
Conv3_2	2×Res. I, k=3, s=1	Conv3_1	(64, 16, 64)
Up3	Deconv., k=3, s=1	Conv1_2	(64, 32, 128)
Conv4_1	Conv., k=3, s=1	Up3	(32, 32, 128)
Conv4_2	2×Res. I, k=3, s=1, no norm	Conv4_1	(32, 32, 128)
Conv5_1	Conv., k=3, s=1, no norm	Conv4_2	(16, 32, 128)
Conv5_2	Conv., k=3, s=1, no norm, no activ	Conv5_1	(3, 32, 128)
Decoder for P_{sun}			
Merge	repeat z_{sun} and concate with M_{sun}	z_{sun}, M_{sun}	(46, 32, 128)
Conv1_1	Conv., k=3, s=1	Merge	(64, 32, 128)
Conv1_2	Res. I, k=3, s=1	Conv1_1	(64, 32, 128)
Conv2_1	Conv., k=3, s=1	Conv1_2	(128, 32, 128)
Conv2_2	Res. I, k=3, s=1	Conv2_1	(128, 32, 128)
Conv3_1	Conv., k=3, s=1	Conv2_2	(64, 32, 128)
Conv3_2	Res. I, k=3, s=1	Conv3_1	(64, 32, 128)
Conv4_1	Conv., k=3, s=1	Conv3_2	(32, 32, 128)
Conv4_2	2×Res. I, k=3, s=1, no norm	Conv4_1	(32, 32, 128)
Conv5_1	Conv., k=3, s=1, no norm	Conv4_2	(16, 32, 128)
Conv5_2	Conv., k=3, s=1, no norm, no activ	Conv5_1	(3, 32, 128)

Table 7. Network architectures of our local appearance encoder-renderer.

Local Appearance Encoder-Renderer			
Name	Layer Description	Input	Output Dim.
P_{sv}	Local lighting maps	–	(3, 64, 128)
z_{local}	Latent code for local content	P_{sv}	(64)
z_{sky}	Latent code for sky light	P_{global}	(16)
z_{sun}	Latent code for sun light	P_{global}	(45)
M_{cos}	Cosine mask for sun light	M_{sun}	(1, 64, 128)
Encoder for z_{local}			
Conv1_1	Conv., k=5, s=1	P_{global}	(32, 64, 128)
Conv1_2	Res. I, k=3, s=1	Conv1_1	(32, 64, 128)
Conv1_3	Conv., k=3, s=2	Conv1_2	(64, 32, 64)
Conv2_1	Conv., k=3, s=1	Conv1_3	(64, 32, 64)
Conv2_2	Res. I, k=3, s=1	Conv2_1	(64, 32, 64)
Conv2_3	Conv., k=3, s=2	Conv2_2	(128, 16, 32)
Conv3_1	Conv., k=3, s=1	Conv2_3	(128, 16, 32)
Conv3_2	Res. I, k=3, s=1	Conv3_1	(128, 16, 32)
Conv3_3	Conv., k=3, s=2	Conv3_2	(128, 8, 16)
Conv4_1	Conv., k=3, s=1	Conv3_3	(128, 8, 16)
Conv4_2	Res. I, k=3, s=1, no norm	Conv4_1	(128, 8, 16)
Conv4_3	Conv., k=3, s=2, no norm	Conv4_2	(64, 4, 8)
Conv5	Conv., k=3, s=1, no norm, no activ	Conv4_3	(64, 2, 8)
Pool	Global Avg. Pool.	Conv5	(64)
Decoder for M_{sil}			
Fc	Linear 64 × 256, and reshape	z_{local}	(8, 4, 8)
Conv1_1	Conv., k=3, s=1	Fc	(32, 4, 8)
Up1	Deconv., k=3, s=1, nearest upsample	Conv1_1	(32, 8, 16)
Conv1_2	2×Res. I, k=3, s=1	Up1	(32, 8, 16)
Conv2_1	Conv., k=3, s=1	Conv1_2	(64, 8, 16)
Up2	Deconv., k=3, s=1, nearest upsample	Conv2_1	(64, 16, 32)
Conv2_2	2×Res. I, k=3, s=1	Up2	(64, 16, 32)
Conv3_1	Conv., k=3, s=1	Conv2_2	(64, 16, 32)
Up3	Deconv., k=3, s=1, nearest upsample	Conv3_1	(64, 32, 64)
Conv3_2	2×Res. I, k=3, s=1	Up3	(64, 32, 64)
Conv4_1	Conv., k=3, s=1	Conv3_2	(32, 32, 64)
Up4	Deconv., k=3, s=1, nearest upsample	Conv4_1	(32, 64, 128)
Conv4_2	2×Res. I, k=3, s=1	Up4	(32, 64, 128)
Conv5_1	Conv., k=3, s=1, no norm	Conv4_2	(16, 64, 128)
Conv5_2	Conv., k=3, s=1, no norm, Sigmoid	Conv5_1	(1, 64, 128)
Renderer for P_{local}			
Concat1	concate z_{local} with z_{sky}	z_{local}, z_{sky}	(80)
Fc	Linear 80 × 1024, and reshape	Concat1	(8, 8, 16)
Conv1_1	Conv., k=3, s=1	Fc	(64, 8, 16)
Conv1_2	2×Res. I, k=3, s=1	Conv1_1	(64, 8, 16)
Up1	Deconv., k=3, s=1	Conv1_2	(64, 16, 32)
Conv2_1	Conv., k=3, s=1	Up1	(128, 16, 32)
Conv2_2	2×Res. I, k=3, s=1	Conv2_1	(128, 16, 32)
Up2	Deconv., k=3, s=1	Conv2_2	(128, 32, 64)
Conv3_1	Conv., k=3, s=1	Up2	(128, 32, 64)
Conv3_2	2×Res. I, k=3, s=1	Conv3_1	(128, 32, 64)
Up3	Deconv., k=3, s=1	Conv3_2	(128, 64, 128)
Conv4_1	Conv., k=3, s=1	Up3	(128, 64, 128)
Conv4_2	2×Res. I, k=3, s=1, no norm	Conv4_1	(128, 64, 128)
Concat2	concate M_{cos} with Conv4_2	Conv4_2, M_{cos}	(129, 64, 128)
Merge	repeat z_{sun} and concate with Concat2	Concat2, z_{sun}	(174, 64, 128)
Conv5_1	Conv., k=3, s=1	Merge	(128, 64, 128)
Conv5_2	2×Res. I, k=3, s=1	Conv5_1	(128, 64, 128)
Conv6_1	Conv., k=3, s=1	Conv5_2	(128, 64, 128)
Conv6_2	2×Res. I, k=3, s=1	Conv6_1	(128, 64, 128)
Conv7_1	Conv., k=3, s=1	Conv6_2	(64, 64, 128)
Conv7_2	2×Res. I, k=3, s=1	Conv7_1	(64, 64, 128)
Conv8_1	Conv., k=3, s=1	Conv7_2	(32, 64, 128)
Conv8_2	2×Res. I, k=3, s=1	Conv8_1	(32, 64, 128)
Conv9_1	Conv., k=3, s=1, no norm	Conv8_2	(16, 64, 128)
Conv9_2	Conv., k=3, s=1, no norm, no activ	Conv9_1	(3, 64, 128)

Table 8. Network architectures of lighting estimator and baselines.

Spatially-varying Lighting Estimator			
Name	Layer Description	Input	Output Dim.
I	Limited-FoV images	–	(3, 240, 320)
l	local pixel positions	–	(2)
F_{global}	extracted global feature	I	(512, 30, 40)
F_{local}	extracted local feature	F_{global}	(128, 60, 80)
F_{pix}	pixel-aligned local feature	F_{local}	(128, 3, 3)
Global Feature Backbone for F_{global}			
Conv1	Conv., k=7, s=1	I	(64, 240, 320)
Conv2	Conv., k=4, s=2	Conv1	(128, 120, 160)
Conv3	Conv., k=4, s=2	Conv2	(256, 60, 80)
Conv4	Conv., k=4, s=2	Conv3	(512, 30, 40)
Conv5	2×Res. I, k=3, s=1	Conv4	(512, 30, 40)
Estimator for M_{vis}			
Conv1	2×Res. I, k=3, s=1	F_{global}	(512, 30, 40)
Up1	Deconv., k=5, s=1, ReLU	Conv1	(256, 60, 80)
Up2	Deconv., k=5, s=1, ReLU	Conv2	(128, 120, 160)
Up3	Deconv., k=5, s=1, ReLU	Conv3	(64, 240, 320)
Conv2	Conv., k=7, s=1, no norm, Sigmoid	Up3	(1, 240, 320)
Estimators for z_{sky} ($c_{\text{out}} = 16$) and z_{sun} ($c_{\text{out}} = 45$)			
Conv1_1	Conv., k=3, s=1	F_{global}	(256, 30, 40)
Conv1_2	Res. I, k=3, s=1	Conv1_1	(256, 30, 40)
Conv2_1	Conv., k=3, s=2	Conv1_2	(128, 15, 20)
Conv2_2	Res. I, k=3, s=1	Conv2_1	(128, 15, 20)
Conv3_1	Conv., k=3, s=2	Conv2_2	(128, 8, 10)
Conv3_2	Res. I, k=3, s=1	Conv3_1	(128, 8, 10)
Conv4_1	Conv., k=3, s=1	Conv3_2	(64, 8, 10)
Conv4_2	Res. I, k=3, s=1	Conv4_1	(64, 8, 10)
Conv5_1	Conv., k=3, s=2, no norm	Conv4_2	(32, 4, 5)
Conv5_2	Conv., k=3, s=1, no norm, no activ	Conv5_1	(c_{out}, 4, 5)
Pool	Global Avg. Pool.	Conv5_2	(c_{out})
Estimators for M_{sun}			
Conv1_1	Conv., k=3, s=1	F_{global}	(256, 30, 40)
Conv1_2	Res. I, k=3, s=1	Conv1_1	(256, 30, 40)
Pool1	Avg. Pool., k=2, s=2	Conv1_2	(256, 15, 20)
Conv2	Conv., k=3, s=1	Pool1	(128, 15, 20)
Pool2	Avg. Pool., k=5, s=5	Conv2	(128, 3, 4)
Fc	reshape and Linear 1536×256	Pool2	(256)
Softmax	Softmax and reshape	Fc	(1, 8, 32)
Estimators for z_{local} ($c_{\text{out}} = 64$) or SH coefficients in Ours_{SH} ($c_{\text{out}} = 108$) or SG lobe axes ($c_{\text{out}} = 48$), lambdas ($c_{\text{out}} = 24$) and weights ($c_{\text{out}} = 72$) in Ours_{SG}			
Extract	3×3 feature patch of F_{pix} pixel-aligned to l	F_{local}, l	(128, 3, 3)
Reshape	flatten	Extract	(1152)
Percp1	Linear 1152×512, and LReLU (0.1)	Reshape	(512)
Percp2	Linear 512×256, and LReLU (0.1)	Percp1	(256)
Percp3	Linear 256×128, and LReLU (0.1)	Percp2	(128)
Percp4	Linear $128 \times c_{\text{out}}$, and LReLU (0.1)	Percp3	(c_{out})

Table 9. Network architectures of stacked hourglass structure.

Stacked Hourglass Structure for F_{local}			
Up	ConvT., k=4, s=2, IN, LReLU (0.1)	F_{global}	(256, 60, 80)
Conv1	Res. I, k=3, s=1	Up	(256, 60, 80)
Hg1	*Hourglass structure of depth=2*	Conv1	(256, 60, 80)
Conv1_t	Res. II, k=3, s=1	Hg1	(256, 60, 80)
Conv1_l	Conv., k=1, s=1, no norm, no activ	Conv1_t	(128, 60, 80)
Conv1_a	Conv., k=1, s=1, no norm, no activ	Conv1_l	(256, 60, 80)
Conv1_b	Conv., k=1, s=1, no norm, no activ	Conv1_t	(256, 60, 80)
Input2	Input2 = Conv1_t+Conv1_a+Conv1_b		(256, 60, 80)
Hg2	Hourglass structure of depth=2	Input2	(256, 60, 80)
Conv2_t	Res. II, k=3, s=1	Hg2	(256, 60, 80)
Conv2_l	Conv., k=1, s=1, no norm, no activ	Conv2_t	(128, 60, 80)

Hourglass Structure of depth=2			
Name	Layer Description	Input	Output Dim.
F	feature for hourglass input	–	(C, H, W)
Conv1_b1	Res. II, k=3, s=1	F	(C, H, W)
Pool1	Avg. Pool., k=2, s=2	F	$(C, H/2, W/2)$
Conv1_b2	Res. II, k=3, s=1	Pool1	$(C, H/2, W/2)$
Conv2_b1	Res. II, k=3, s=1	Conv1_b2	$(C, H/2, W/2)$
Pool2	Avg. Pool., k=2, s=2	Conv1_b2	$(C, H/4, W/4)$
Conv2_b2	Res. II, k=3, s=1	Pool2	$(C, H/4, W/4)$
Conv2_b2p	Res. II, k=3, s=1	Pool2	$(C, H/4, W/4)$
Conv2_b3	Res. II, k=3, s=1	Conv2_b2p	$(C, H/4, W/4)$
Up2	Deconv., no conv, no activ	Conv2_b3	$(C, H/2, W/2)$
Output2	Output2 = Conv2_b1 + Up2		$(C, H/2, W/2)$
Conv1_b3	Res. II, k=3, s=1	Output2	$(C, H/2, W/2)$
Up1	Deconv., no conv, no activ	Conv1_b3	(C, H, W)
Output1	Output1 = Conv1_b1 + Up1		(C, H, W)
Output	output of the hourglass	Output1	(C, H, W)

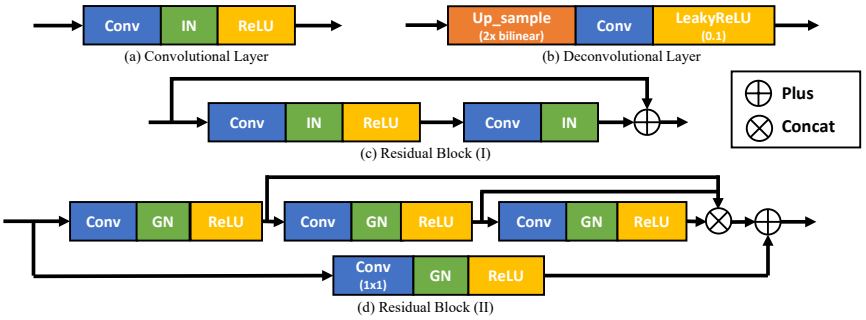


Fig. 22. Common blocks used in our network architectures (with default settings).

References

1. ambientCG, public domain materials for physically based rendering. [licensed under CC0 1.0 universal], https://ambientcg.com 17
2. Blender., https://www.blender.org 17, 18
3. Hold-Geoffroy, Y., Athawale, A., Lalonde, J.F.: Deep sky modeling for single image outdoor lighting estimation. In: Proc. of Computer Vision and Pattern Recognition (2019) 18, 20, 22
4. Kalantari, N.K., Ramamoorthi, R.: Deep high dynamic range imaging of dynamic scenes. ACM Transactions on Graphics (Proc. of ACM SIGGRAPH) **36**(4) (2017) 18
5. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: Proc. of International Conference on Learning Representations (2015) 18
6. Lalonde, J.F., Asselin, L.P., Becirovski, J., Hold-Geoffroy, Y., Garon, M., Gardner, M.A., Zhang, J.: The laval HDR sky database. [free license for academic or government-sponsored researchers] (2016), http://sky.hdrdb.com 18
7. Li, Z., Shafiei, M., Ramamoorthi, R., Sunkavalli, K., Chandraker, M.: Inverse rendering for complex indoor scenes: Shape, spatially-varying lighting and SVBRDF from a single image. In: Proc. of Computer Vision and Pattern Recognition (2020) 20
8. Liu, D.C., Nocedal, J.: On the limited memory BFGS method for large scale optimization. Mathematical Programming **45**(1-3), 503–528 (1989) 20
9. Otsu, N.: A threshold selection method from gray-level histograms. IEEE Transactions on Systems, Man, and Cybernetics **9**(1), 62–66 (1979) 18
10. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: PyTorch: An imperative style, high-performance deep learning library. In: Proc. of Neural Information Processing Systems (2019) 18, 20
11. Yu, P., Guo, J., Huang, F., Zhou, C., Che, H., Ling, X., Guo, Y.: Hierarchical disentangled representation learning for outdoor illumination estimation and editing. In: Proc. of International Conference on Computer Vision (2021) 19
12. Zamir, A.R., Shah, M.: Image geo-localization based on multiple nearest neighbor feature matching using generalized graphs. IEEE Transactions on Pattern Analysis and Machine Intelligence **36**(8), 1546–1558 (2014) 22
13. Zhu, Y., Zhang, Y., Li, S., Shi, B.: Spatially-varying outdoor lighting estimation from intrinsics. In: Proc. of Computer Vision and Pattern Recognition (2021) 17, 19, 20, 21, 22