ISSN 1213-6972 (print)
ISSN 1213-6964 (online)
Journal of WSCG
http://www.wscg.eu
Vol.31, No-1-2, 2023

Real-time Light Estimation and Neural Soft Shadows for AR Indoor Scenarios

Alexander Sommer[1]
alexander.sommer@hs-rm.de

Ulrich Schwanecke[1]
ulrich.schwanecke@hs-rm.de

Elmar Schoemer[2]
schoemer@uni-mainz.de

[1] Computer Vision and Mixed Reality Group, RheinMain University of Applied Sciences Wiesbaden Rüsselsheim, Germany
[2]Institute of Computer Science, Johannes Gutenberg University Mainz, Germany

ABSTRACT

We present a pipeline for realistic embedding of virtual objects into footage of indoor scenes with focus on real-time AR applications. Our pipeline consists of two main components: A light estimator and a neural soft shadow texture generator. Our light estimation is based on deep neural nets and determines the main light direction, light color, ambient color and an opacity parameter for the shadow texture. Our *neural soft shadow* method encodes object-based realistic soft shadows as light direction dependent textures in a small MLP. We show that our pipeline can be used to integrate objects into AR scenes in a new level of realism in real-time. Our models are small enough to run on current mobile devices. We achieve runtimes of 9ms for light estimation and 5ms for neural shadows on an iPhone 11 Pro.

Keywords: augmented reality, light estimation, shadow rendering, neural soft shadows

1 INTRODUCTION

We propose a method for realistically inserting virtual objects into indoor scenes in the context of augmented reality applications. Thereby we first estimate the current lighting situation in the scene from a single RGB image captured by the camera of, for example, a mobile device. Then we use this information to insert the virtual object into the existing scene as plausibly and realistically as possible.

The light situation in an existing scene can be captured by placing a light probe at the position of the image. This can create a 360° high-dynamic range (HDR) panorama, also called environment map, of the scene. Such an HDR image contains a large amount of information about bright and dark areas that would be clipped as black or white in an ordinary 8Bit low-dynamic range (LDR) image. Since the map contains information about the illumination of each direction of the scene at a given point, this environment map can be utilized to illuminate an object as if it were in the scene using methods from image-based lighting (IBL) [8]. Some techniques have been developed to estimate this environment map from a single limited field-of-view LDR image without additional 360° cameras using neural nets and deep learning [12, 28, 27]. However, such an environment map is only valid for a single specific point in the scene. Moreover IBL techniques can be used to realistically illuminate objects with spatial varying light. Shadows in IBL are created by tracing the path of light and its interaction with other objects in the scene. While this produces a very realistic shadow cast, a large number of path traces is required. This is computationally intensive and therefore not suitable for real-time applications on mobile devices.

Figure 1: Example application of our pipeline: The light estimation determines the light direction and ambient color for rendering the inserted object. Based on the determined light direction, additional neural soft shadows are generated to create a realistic shadow cast as texture.

Alternatively, parametric models exist that describe the light sources as physical objects in a 3D scene. In contrast to an environment map from IBL, these models are valid for the entire scene. In the simplest case, such a parametric model can be for example a directional light with a fixed direction. Parametric light sources have a long history in computer graphics and there are several methods to efficiently calculate the shadow cast by objects. However, they often have to be modeled manually by a 3D artist for an existing scene. Recently, methods came up to estimate parametric lights directly from an input image using neural networks [5, 13, 11]. Our work takes up on these methods. We restrict ourselves to reliably predict the main light direction at a given point from a limited field-of-view LDR indoor RGB camera image and additionally determine the light color as well as the ambient color.

Especially in indoor scenes, the lighting situation is very complex. For the realistic overlay of virtual objects in the context of AR [26], it makes a big difference whether a realistic or visual convincing shadow cast is present. Many other light estimation works map the existing lighting situation, but are only able to realistically insert virtual objects through offline rendering, e.g. ray tracing. Most shadows in indoor scenes are soft shadows as they are caused by light objects in a relatively short distance with a certain surface area. They are much more complex to compute than hard shadows caused by a quasi infinitely distant light source like the sun in outdoor scenes. We present a method to use the estimated light direction from the previous part to generate realistic indoor soft shadows in real-time. For this purpose we present a novel approach to encode precomputed ray-traced soft shadows using a neural network. This small network can be queried in real-time to generate a shadow texture depending on the light direction (see Fig. 1).
Our *main contributions* are as follows:

1. An improved deep neural network for parametric light direction estimation in indoor scenes.

2. A new method for encoding shadow textures in an MLP that is memory friendly and fast to query.

3. A complete pipeline for light estimation and shadow creation for real-time AR applications on mobile devices.

2 RELATED WORK

Existing work related to ours can be roughly divided into two categories. On the one hand, research in the area of estimating the existing light situation in the real scenery and, on the other hand, research on how to use this information for the realistic insertion of virtual objects into the augmented reality.

Light estimation is a classical problem in the field of computer vision or computer graphics as a subarea of 3D scene reconstruction. An accurate determination of the existing lighting conditions is crucial for a convincing insertion of virtual objects into the real environment.

Classic approaches usually require multiple images and/or more detailed knowledge about the underlying scene geometry. For example, Debevec and Malik showed how the omnidirectional HDR radiance map can be reconstructed using multiple shots of a reflecting sphere with different exposure settings [9] and how to render synthetic objects into real scenes [8]. Lombardi and Nishino [19] showed how illumination can be reconstructed from a single image of an object with known geometry. Balcı and Güdükbay [2] reconstructed illumation based on the shadows in scenes that were mainly illuminated by the sun. Baron and Malik [3] reconstructed not only the illumination but also the geometry and reflectivity of an unknown object from an image using shape priors. Lopez-Moreno et al. [20] presented an approach based on heuristics that does not require geometric knowledge.

With the rise of machine learning based approaches the need for information about the scenery could be further reduced. There exist quite some work that estimate lighting information and environment maps. For example, Hold-Geoffroy et al. [14] used a deep neural net to predict the illumination in outdoor scenes from a single image by relying on a physically-based sky model. Gardner et al. [12] estimated an HDR illumination map for indoor scenes also from a single image by splitting the process into light position estimation and HDR intensity estimation. Song and Funkhouser [28] used a multi-stage approach to predict a 360° LDR map from a single image and completed geometry and intensity on HDR scale. Somanath and Kurz [27] predicted a true HDR map from a single camera image in a single stage approach tailored to mobile augmented reality (AR) real-time applications. Other approaches focus more on estimating light in form of low dimensional parameters. Garon et al. [13] used spherical harmonic coefficients as light model. Cheng et al. [5] also used spherical harmonics for their light model, but used the images from the front and rear camera for the estimation.

Gardner et al. [11] described a deep neural net that estimates light parameters for individual light objects. This method is the closest to our work. They used the Laval Indoor HDR dataset [12] which contains about 2100 HDR maps to train the network. These parameters for the training data were determined by fitting ellipses on the HDR intensity maps. The brightest area in the map was detected and the ellipse then was fitted by region growing. This area was masked and the process was repeated to determine a number of light sources.

The parameters of the light source were defined by the size of the ellipse, average HDR intensity in the ellipse area and average HDR color value. Furthermore, a predicted depth map was used to determine the distance to the light source. We also use the Laval Indoor HDR dataset and with a DenseNet pretrained on ImageNet a similar network architecture. However, unlike Gardner et al. we estimate a light direction and therefore do not need to rely on predicted depth maps for the dataset.

Shadow calculation is a very broad and relatively old field of research in computer graphics. It ranges from simple methods for computing hard shadows, such as projection shadows [4], shadow mapping [31] and shadow volumes [7] to more advanced methods for computing soft shadows like image-based soft shadows [1], geometry-based soft shadows [22] and volumetric shadows [18].

In contrast to previous work, we present a new approach in which we encode pre-computed shadow textures for an object in the weights of a neural network. This has the advantage that realistic soft shadows can be displayed in real-time on mobile devices, since the network can be queried very quickly. The idea of encoding images or textures in neural networks is not new. Stanley [29] encoded image information in Compositional Pattern Producing Network (CPPN) inspired by encoding in natural DNA. Rainer et al. [25, 24] used neural networks to compress the bidirectional texture function (BTF). Mildenhall et al. [21] trained an multilayer perceptron (MLP) to generate novel views from unknown perspectives of complex scenes. They used a mapping for the input coordinates to create a higher dimensional input space that allowed more high frequency variations in their output. This strategy was inspired by the positional encoding in the Transformer architecture [30] and is also used by our method.

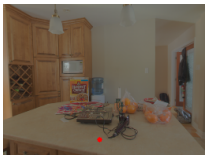
3 LIGHT ESTIMATION

To estimate the existing light situation in a scene using a single RGB image, we characterize the light situation by a set of parameters

$$(\boldsymbol{d}, \boldsymbol{c}, \boldsymbol{a}, o)\,. \tag{1}$$

Here $\boldsymbol{d} \in \mathbb{R}^3$ is a unit vector which determines the light direction, $\boldsymbol{c} \in \mathbb{R}^3$ is the light color defined by RGB values with normalized components in $[0,1]$ and $\boldsymbol{a}$ is an RGB vector corresponding to the ambient lighting of the scene. The parameter o is a scalar value and measure for the opacity of the shadow texture described in Section 4. A value $o = 1$ corresponds to an alpha value of 100% and a value of $o = 0$ corresponds to an alpha value of 0%, i.e. invisible shadows. We train a convolutional neural network (CNN) to predict these parameters from a single RGB image with a resolution of 256x192 pixels.

(a) Original panorama



(b) Cropped image

(c) Warped panorama

Figure 2: For a given panorama (a), the image information from inside the red frame is used to create the rectified cropped image (b). A warped panorama (c) is projected around the insertion point (red point in (b)).

3.1 Input Data

For training the network, a large number of images is needed for which the exact light situation of the whole scene is known. 360° HDR panoramas are particularly suitable for this, since one can crop a limited field-of-view image from them to obtain input images (see Fig. 2a & Fig. 2b), while still being able to recover the entire lighting situation of the scene. We use the Laval Indoor HDR dataset [12] which contains about 2100 HDR panoramas, taken at different indoor scenes.

Like Gardner et. al [12], we extract 8 different limited field-of-view crops per panorama at random polar angles θ between 60° and 120° and azimuth angles ϕ between 0° and 360°. We use a field-of-view (FOV) of 85° to approximate the viewing angle of non-wide-angle cameras in modern smartphones. We perform a rectilinear projection (see red frame in Fig. 2a)) to back-project the distortion in the panoramas. The 360° HDR panorama describes the light situation of the whole scene at the point where the camera was placed for the panorama. However, this does not correspond to the exact lighting situation around the cropped image. For finding out the exact light situation at that point, one would have to shoot a new 360° HDR panorama at the virtual camera location of the cropped image. To estimate the light situation at this location we rotate the original panorama so that the cropped area is exactly in the center and then apply the same warping operator as described in [12]. The resulting new panorama (see Fig. 2c) is an approximation of the panorama around the virtual camera location of the cropped area.

We use each of the warped panoramas to extract the light parameters for the corresponding cropped input

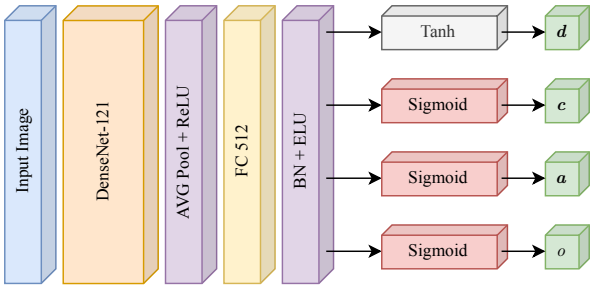


Figure 3: Proposed light estimation network architecture.

image. We first determine the pixel intensity I_{ij} by adding the individual RGB channels with weights that correspond to the natural perception of the individual colors, i.e.

$$I_{ij} = 0.0722 \cdot R_{ij} + 0.7152 \cdot G_{ij} + 0.2126 \cdot B_{ij}, \quad (2)$$

where i is the pixel's column and j its row.

Then we mask the areas where the intensity is greater than 5% of the maximum intensity $I_{\max}$ as highlights. It should be noted that this is only applicable when working with HDR data. To determine the average light direction from the highlight area, we introduce two weights. First, the light direction of each pixel is weighted by its intensity. Second, the light direction of each pixel is weighted by the area that this pixel occupies on the unit sphere:

$$\omega_{ij} = \frac{2\pi^2}{w \cdot h} \sin\left(\frac{j+0.5}{h}\pi\right) \quad (3)$$

where j is the pixel's row and w, h are the width and height of the panorama. This is necessary because, for example, an area near the poles occupies significantly more pixels on the panorama than an area with the same size at the equator. The resulting average light direction is the parameter $\boldsymbol{d}$. To determine the light color $\boldsymbol{c}$, the same weights are applied to the individual RGB values of the highlight area in a tone-mapped version of the panorama to obtain a mean highlight color. The ambient color $\boldsymbol{a}$ can be determined from the remaining pixel values of the tone-mapped panorama by using the same procedure. We determine the value for the opacity parameter o from the quotient of the summed weighted intensities for the highlight areas I_l^{tot} and analog for the remaining areas I_a^{tot}:

$$o = 1 - \tanh\left(\frac{I_a^{\text{tot}}}{0.05 \cdot I_l^{\text{tot}}}\right). \quad (4)$$

The less the intensities from the highlight areas differ from those of the ambient area, the lower the opacity of the shadow textures.

3.2 Network Architecture

As mentioned before we use a CNN to estimate the parameters from the input RGB image. Since the dataset is too small to train a network from scratch, we use a DenseNet-121 [17], pretrained on ImageNet [10] as an encoder. The block configuration is (6, 12, 24, 16) with a growth rate of 32, a compression of 0.5 and a batch norm size of 4. Furthermore, 64 initial features, ReLU activations and 2D average pooling with a pool size of 4 are used. The classifier of the DenseNet is removed, so the network produces a latent vector with size 512. This is forwarded to a fully connected (FC) 512 layer with batch norm and ELU activation. For each of the four parameters there is a separate FC layer as network head. The heads for the parameters $\boldsymbol{c}$, $\boldsymbol{a}$, o are each normalized using a sigmoid function so that they lie between $(0,1)$. For the parameter $\boldsymbol{d}$ we use a tanh activation function and normalize the entire vector to unit length. The complete architecture is visualized in Figure 3.

Metric	Gardner19(1)	Ours
RMSE	0.1114	**0.1101**
si-RMSE	0.1518	**0.1501**
RMLE	0.07007	**0.06928**
Angular Error	3.556°	**3.542°**

Table 1: Comparision by different widely used metrics of our method with the state of the art parametric indoor light estimation by Gardner et al. [11] with one light source. Best results in bold.

3.3 Training & Implementation

During training, we directly compare the estimated parameters with the ground truth parameters. Thereby individual losses for each head are calculated as mean squared error. The total loss function is the weighted sum of the individual losses, i.e.

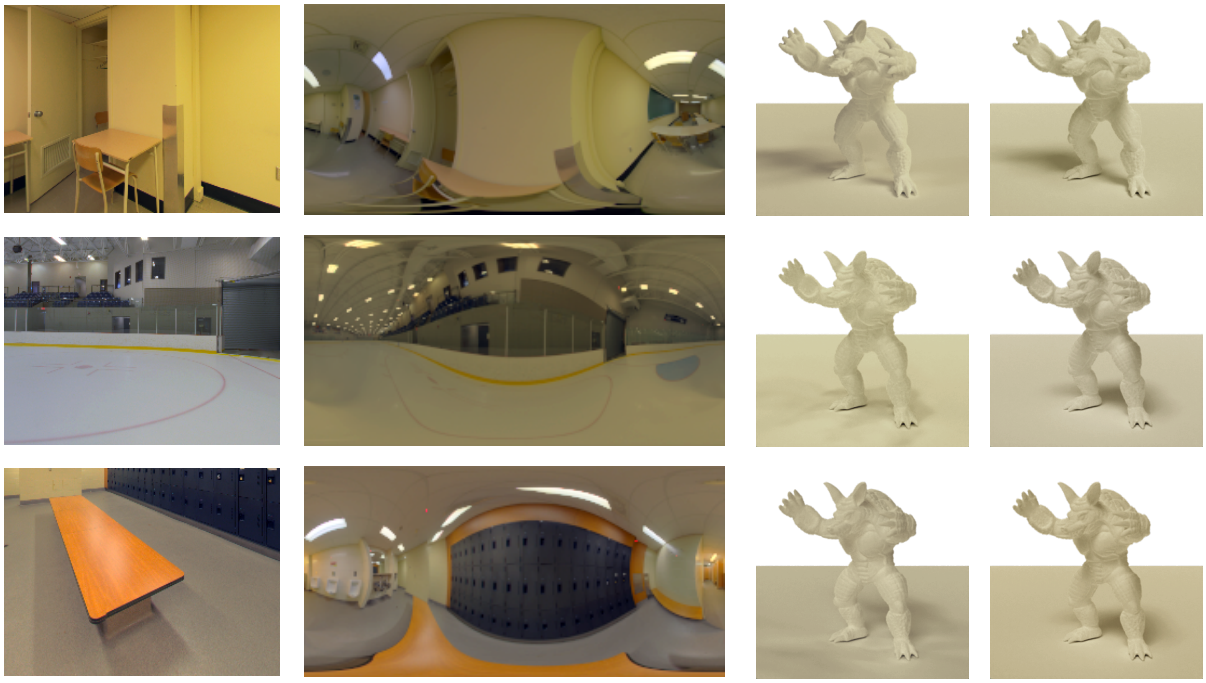
$$\begin{aligned}\mathscr{L} &= \omega_d l_2(\boldsymbol{d}^{\text{est}}, \boldsymbol{d}^{\text{gt}}) + \omega_c l_2(\boldsymbol{c}^{\text{est}}, \boldsymbol{c}^{\text{gt}}) \\ &\quad + \omega_a l_2(\boldsymbol{a}^{\text{est}}, \boldsymbol{a}^{\text{gt}}) + \omega_o l_2(o^{\text{est}}, o^{\text{gt}}).\end{aligned} \quad (5)$$

We weight the individual losses differently with the weights $\omega_d = 5$, $\omega_c = 2$, $\omega_a = 2$ and $\omega_o = 1$. Since a correct estimation of the direction is of utmost importance for us, ω_d gets the highest value.

We train the network for a total of 60 epochs using an Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The learning rate $l_r = 0.001$ is halved every 15 epochs. We use a batch size of 128 samples and a random 85/15 split of the dataset for training and validation. Scenes unknown to the network were used for testing. Typically, training takes about 2 hours on two Nvidia RTX A6000 GPUs. In total, our network consists of 7.7M parameters. The interference time on the iPhone 11 Pro GPU is 62ms and on the Apple Neural Engine (ANE) 9ms.

3.4 Evaluation

Comparing the results of two light estimation approaches is challenging. Since qualitative evaluation



(a) Input image (b) GT HDR panorama (c) GT result (d) Our result

Figure 4: Exemplified representation of our evaluation. (a) shows the input image, (b) the corresponding GT HDR panorama, (c) the GT image of the Armadillo rendered with IBL techniques, and (d) the image of the same Armadillo rendered using the light parameters from our light estimation.

always contains personal bias, we rely on a purely quantitative measure for this evaluation. We don't compare our method with approaches, that do not estimate a parametric light direction but spatially varying light coefficients like spherical harmonics [5, 13] or complete environment maps [12, 28, 27], because we especially need the light direction for the shadow calculation (Sec. 4). We therefore compare our light estimation approach only with the work of Gardner et al. [11] when using one main light source. We neglect our opacity parameter o at this point, since its use is mainly for the shadow textures presented in Section 4 and will be evaluated in the overall pipeline evaluation in Section 5.

We use a simple scene with an armadillo and a plane as a shadow catcher (see Fig. 4c). For a given input image (see Fig. 4a), we render a GT image (see Fig. 4c) with the corresponding warped GT environment map (Fig. 4b), as described in Section 3.1, with IBL techniques. We then estimate light parameters with the respective light estimation. The same scene is rendered again with a parametric light source and ambient color (see Fig. 4d).

To compare renderings of the two predictions with the GT image we use 4 different metrics. On the one hand RMSE as well as the scale-invariant si-RMSE and RMLE and on the other hand a per pixel RGB angular error [15]. The standard RMSE is a good measure for the error in the relation between ambient and light intensity. The two scale-invariant measures filter out differences in the scales of the two images and are therefore good measures for errors in light position due to difference in shadows. The RGB angular error, on the other hand, comes from whitebalance research and is a good measure to evaluate the color predicition of the light source and the ambient color.

In total, we evaluated 977 images from a test set unknown to the network. We used Blender [6] for all renderings. Table 1 shows the results of our evaluation. It can be seen that our method performs 1-1.2% better than the previous state-of-the-art method in all metrics when considering only a single light source.

4 SHADOWS

We aim to generate a planar shadow texture (see Fig. 5b), i.e. a 2D grayscale image, depending on the light direction defined by a unit vector $\boldsymbol{d} \in \mathbb{R}^3$ for a specific object (see Fig. 5a). Our experiments showed that the use of cartesian coordinates leads to a more stable training than spherical coordinates since the network seems to have problems with the discontinuity between $\phi = 2\pi$ and $\phi = 0$. This results in a *shadow function*

$$f : \mathbb{R}^5 \longrightarrow \mathbb{R}, \quad f(i, j, \boldsymbol{d}) \longrightarrow v, \tag{6}$$

(a) Object (b) Shadow texture

Figure 5: A chair lit by a front light (a) with the corresponding shadow texture (b).

Figure 6: Shadow textures of the Armadillo (see Fig. 4c) from different light directions $\boldsymbol{d}$.

that maps pixel position (i, j) together with a light direction $\mathbf{d}$ to a grayscale value v. We use a MLPs are a universal function approximator [16], to represent the desired shadow function.

4.1 Input Data

We train one specific network for each individual model. As training data, we use shadow textures for a variety of different light directions. These textures are created with a simple scene setup and the Cycles render engine in Blender [6]. The scene consists of a quadratic plane that acts as a shadow catcher. The plane is dimensioned so that its side length is three times as long as the largest side of the bounding box that contains the object to be trained. The object is centered on the plane and is assigned a material that is invisible to the render engine but allows shadow casting. An orthographic camera from the top view captures the textures. A directional light (sun in blender) with an opening angle of 20° is used as the light source. This type of light is defined by one direction and still produces soft shadows. It's therefore well suited as an approximation for indoor shadows. This light source is set to different light directions for the individual training samples. We use uniformly distributed spherical angles θ, ϕ. Where θ takes values from 0° to 45° with an increment of 4.5° and ϕ takes values from 0° to 360° with an increment of 12°. This results in a total of 301 texture samples. For each sample, we use a resolution of 256x256 pixels. Figure 6 shows an example of shadow textures for different light directions for the Armadillo (see Fig. 4c).

4.2 Network Architecture

As mentioned before (see Eq. (6)), all information about the shadows is mapped by pixel-wise functions from 5D space to 1D grayscale information. Since neural networks tend to learn a low-frequency bias, we assist them in learning high-frequency details by mapping the 5D input to a higher dimensional space, as shown by Rahaman et al. [23]. This technique is also used very successfully with NeRFs [21]. Similar to Vaswani et al. [30] with Transformers, we use an encoder function Φ to map each of the five input dimensions $x \in \mathbb{R}$ to a higher dimensional sequence of alternating sine and cosine functions:

$$\begin{aligned}\Phi(x) = \big(&\sin\big(2^0\pi x\big), \cos\big(2^0\pi x\big), \ldots, \\ &\ldots, \sin\big(2^{L-1}\pi x\big), \cos\big(2^{L-1}\pi x\big)\big)\end{aligned} \tag{7}$$

where L is a dimensionality parameter. The image space (i, j) is normalized to values in [0,1]. For the image space encoding is done with a dimensionality parameter $L = 10$. The elements of the light direction vector $\boldsymbol{d}$ by definition take only values in [-1,1] and for their encoding we choose an $L = 4$ analogous to the viewing direction vector in [21]. In total we map the $\mathbb{R}^5$ input space to higher dimensional space of $\mathbb{R}^{64}$. The input passes through h hidden layers, each with a filter size s, and is activated with ReLUs after each hidden layer (see Fig. 8). In our experiments we use a filter size s of 128 to 256 and a number of hidden layers h from 1 to 4. The output value v of the network is normalized with a sigmoid function between 0 and 1.

4.3 Training

During training, for each shadow texture sample k with fixed light direction $\boldsymbol{d}$, we take a number of N random continuous pixel locations $(i, j) \in [0, 1]$. Here, the ground truth grayscale value $v^{\text{gt}}_{i,j}$ at the continuous location (i, j) is obtained by bilinear interpolating from the known values at the discrete surrounding known pixel values. It should be mentioned that it is also possible to train the network without interpolation only on random known discrete pixel values. This speeds up the training by a factor of 5 since filtering is a bottleneck. On the other hand, it reduces the quality of the network, and the ability to predict different resolutions with the

(a) Ground truth

(b) Ours

Figure 7: An example of our qualitative evaluation. Left: Coffee table rendered with the ground truth HDR panorama around the insertion point. Right: Coffee table rendered with a directional light and ambient color from our light estimation and shadow texture from our neural soft shadows.

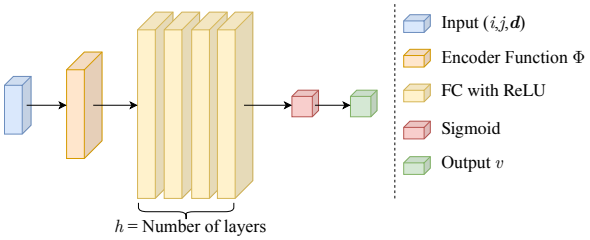


Figure 8: Proposed shadow network architecture.

network is lost. As loss function $\mathscr{L}$ we take the mean squared error loss l_2 between estimated pixel value v^{est} and interpolated pixel value v^{gt}:

$$\mathscr{L} = l_2(v^{\text{est}}, v^{\text{gt}}). \tag{8}$$

4.4 Implementation

One advantage of our method is that the resulting network is very small and thus not only requires little memory, but a forward pass also has a low interference time. The forward passes for all 65536 pixels of a 256x256 texture need in total about 33ms on the GPU of the iPhone 11 Pro and 5ms on the ANE. Assuming a filter size $s = 128$ and a number of hidden layers $h = 3$ the network has just 58k parameters. The data set with its 301 grayscale images with a resolution of 256x256 is small enough to be loaded completely into the memory even with simple consumer GPUs. We train our network for a total of 10000 epochs and need about 5 minutes (or just under a minute without bilinear filtering) on an Nvidia RTX A6000. As in Section 3.3, we again use an Adam optimizer with standard values of $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We apply an exponential learning rate decay ($\gamma = 0.99977$) to the initial learning rate $l_r = 0.001$ so that it is reduced to one-tenth of the original value after 10000 epochs. Per texture sample we use $N = 256$ pixel locations, which results in 77k network passes per epoch.

	GT	Ours
Rating	3.49 ± 0.38	3.26 ± 0.46
Votes	0.544%	0.456%

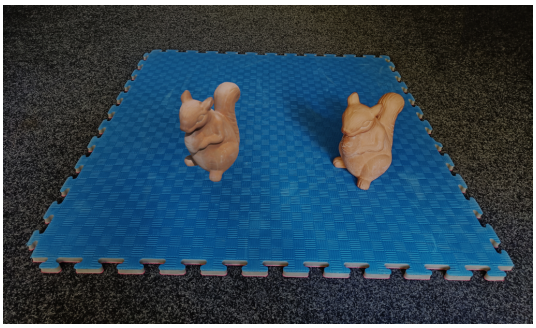
Table 2: Results of the qualitative evaluation (20 images, 50 participants). Rating describes how realistically an objects fits into the scene considering only lighting and shadows on a scale from 1 (very unrealistic) to 5 (very realistic). Votes denotes the percentage of which image was prefered in terms of realistic look (50% = perfect confusion).

4.5 Limitations

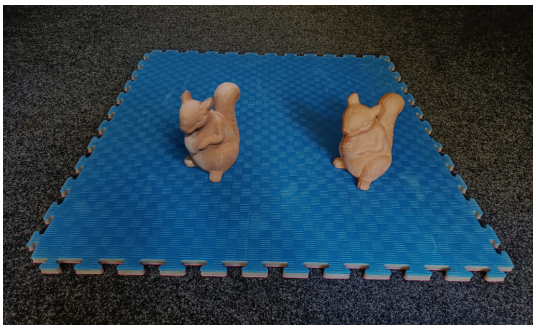
Currently, our method is only suitable for creating a planar shadow texture for the plane it sits on. This is sufficient for of AR applications, where an object is placed in the middle of an empty room and is far enough away from walls to cast a shadow on them. Problems arise when a virtual object should cast shadows on another virtual object or on non-virtual objects in the scene.

5 OVERALL PIPELINE EVALUATION

We determine the overall quality of our entire pipeline with a qualitative evaluation. For this we use new HDR panoramas that are not from the Laval Indoor HDR dataset and have not yet been seen by the network. For each panorama we choose a cropped rectified image where a virtual object should be inserted. We use the light estimation from Section 3 to determine the light direction, light color, ambient color and the opacity value for the shadows. We then use the light direction to determine the shadow texture using our method from Section 4. We insert the object into the image



(a) Without shadow cast



(b) With neural shadow texture

Figure 9: Comparison between a real clay squirrel (right) and the virtual object (left) rendered with the light parameter of the light estimation from Section 3. (a) shows the object without shadow cast and (b) with the neural shadow texture from our method in Section 4.

and render it using only a directional light and ambient lighting. We also add the neural shadow texture with the estimated opacity (see Fig. 7a). In comparison, we determine the warped panorama (see Sec. 3.1) at the insertion point and render the same object with ray traced IBL and a plane as shadow catcher (see Fig. 7b).

A total of 20 images (see supplementary material) were created for qualitave evaluation. We showed these images to 50 participants. On the one hand, the participants were asked to assess how realistically an object fits into the existing scene in terms of its lighting and shadows. For the rating, we use the Likert scale with values from 1 (very unrealistic) to 5 (very realistic). Explicitly the participants were told not to consider syle, proportions, object selection and context. On the other hand, the participants were shown both pictures (see Fig. 7) next to each other and they were asked to decide which of the two pictures they thought was more realistic looking in terms of lighting and shadows. Table 2 shows the results of our survey. It turns out that the participants as a whole give the ground truth visualizations only a slightly higher quality rating than our visualizations. This is also confirmed by the fact that quite a few participants prefer our visualization to the ground truth in a direct comparison.

Furthermore, in Figure 9 we compare a real object with a rendered virtual version. For this we place a real clay squirrel in the room and leave space for the virtual version. The photo was taken with an ordinary smartphone and the light estimation from Section 3 was used to determine the light direction, light color, ambient color and the opacity value of the shadows. The virtual squirrel was inserted on the left and rendered with the light parameters. Figure 9a shows the virtual squirrel without shadow cast. Figure 9b shows the squirrel with the neural soft shadow texture generated with our method from Section 4. It is easy to see that without shadows the object looks out of place in the scene. The subtle soft shadow of our method, on the other hand, conveys immersion.

6 CONCLUSION

We presented a complete pipeline for realistic embedding of virtual objects into indoor scenes. Our light estimation determines a parametric description of the light situation from an RGB image as input. Our neural soft shadow method generates realistic soft shadows as textures that allow to embed virtual objects in previously unknown levels of realismn in real-time into AR scenes. Of course, our method is not suitable for reproducing complex lighting situations exactly, but it is suitable for giving the viewer a convincing sense of immersion. This is supported by our user test where approximately the same number of subjects preferred our method over ground truth visualization. Our entire pipeline is real-time capable on current mobile devices.

In particular, our fundamental work in the area of neural soft shadows opens up a wide range of possibilities for future research. At the moment we are working on how to effectively transfer our method to the shadow cast on walls. In this case, the distance to the wall adds another degree of freedom to the problem. It would be interesting to incorporate more complex light sources, such as area lights, with further parameters like light size in neural shadows. It is also exciting to see if multiple light sources can be represented as neural soft shadows. Furthermore, we could imagine that complex shadows of semi-transparent objects could be another future application of our method.

ACKNOWLEDGMENTS

This project (HA project no. 1102/21-104) is financed with funds of LOEWE - Landes-Offensive zur Entwicklung Wissenschaftlich-ökonomischer Exzellenz, Förderlinie 3: KMU-Verbundvorhaben (State Offensive for the Development of Scientific and Economic Excellence).

REFERENCES

[1] Maneesh Agrawala, Ravi Ramamoorthi, Alan Heirich, and Laurent Moll. Efficient image-based methods for rendering soft shadows. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 375–384, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[2] Hasan Balcı and Uğur Güdükbay. Sun position estimation and tracking for virtual object placement in time-lapse videos. *Signal, Image and Video Processing*, 11, 07 2017.

[3] Jonathan T. Barron and Jitendra Malik. Shape, illumination, and reflectance from shading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37:1670–1687, 2013.

[4] James F. Blinn. Me and my (fake) shadow. *IEEE Computer Graphics and Applications*, 8:82–86, 1988.

[5] Dachuan Cheng, Jian Shi, Yanyun Chen, Xiaoming Deng, and Xiaopeng Zhang. Learning scene illumination by pairwise photos from rear and front mobile cameras. *Computer Graphics Forum*, 37:213–221, 10 2018.

[6] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2022.

[7] Franklin C. Crow. Shadow algorithms for computer graphics. In *Proceedings of the 4th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '77, pages 242–248, New York, NY, USA, 1977. Association for Computing Machinery.

[8] Paul Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 189–198, New York, NY, USA, 1998. Association for Computing Machinery.

[9] Paul E. Debevec and Jitendra Malik. Recovering high dynamic range radiance maps from photographs. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '97, pages 369–378, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[11] Marc-André Gardner, Yannick Hold-Geoffroy, Kalyan Sunkavalli, Christian Gagné, and Jean-François Lalonde. Deep parametric indoor lighting estimation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7174–7182, 10 2019.

[12] 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 Trans. Graph.*, 36(6), nov 2017.

[13] Mathieu Garon, Kalyan Sunkavalli, Sunil Hadap, Nathan Carr, and Jean-Francois Lalonde. Fast spatially-varying indoor lighting estimation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6901–6910, 06 2019.

[14] Yannick Hold-Geoffroy, Kalyan Sunkavalli, Sunil Hadap, Emiliano Gambaretto, and Jean-François Lalonde. Deep outdoor illumination estimation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2373–2382, 07 2017.

[15] S.D. Hordley and G.D. Finlayson. Re-evaluating colour constancy algorithms. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 1, pages 76–79 Vol.1, 2004.

[16] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Netw.*, 2(5):359–366, jul 1989.

[17] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2017.

[18] James T. Kajiya and Brian P Von Herzen. Ray tracing volume densities. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '84, pages 165–174, New York, NY, USA, 1984. Association for Computing Machinery.

[19] Stephen Lombardi and Ko Nishino. Reflectance and illumination recovery in the wild. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(1):129–141, jan 2016.

[20] Jorge Lopez-Moreno, Elena Garces, Sunil Hadap, Erik Reinhard, and Diego Gutiérrez. Multiple light source estimation in a single image. *Computer Graphics Forum*, 32, 12 2013.

[21] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM*, 65(1):99–106, dec 2021.

[22] Steven Parker, Peter Shirley, and Brian Smits. Single sample soft shadow. *Tech. Rep. UUCS-98-019*, 10 1998.

[23] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Dräxler, Min Lin, Fred A. Hamprecht, Yoshua Bengio, and Aaron C. Courville. On the spectral bias of neural networks. In *ICML*, 2018.

[24] Gilles Rainer, Abhijeet Ghosh, Wenzel Jakob, and Tim Weyrich. Unified neural encoding of btfs. *Computer Graphics Forum*, 39:167–178, 05 2020.

[25] Gilles Rainer, Wenzel Jakob, Abhijeet Ghosh, and Tim Weyrich. Neural btf compression and interpolation. *Computer Graphics Forum*, 38:235–244, 05 2019.

[26] Kai Rohmer, Wolfgang BÃ¼schel, Raimund Dachselt, and Thorsten Grosch. Interactive near-field illumination for photorealistic augmented reality on mobile devices. In *2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 29–38, 2014.

[27] Gowri Somanath and Daniel Kurz. Hdr environment map estimation for real-time augmented reality. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11293–11301, 06 2021.

[28] Shuran Song and Thomas Funkhouser. Neural illumination: Lighting prediction for indoor environments. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6911–6919, 06 2019.

[29] Kenneth O. Stanley. Compositional pattern producing networks: A novel abstraction of development. *Genetic Programming and Evolvable Machines*, 8(2):131162, jun 2007.

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc.

[31] Lance Williams. Casting curved shadows on curved surfaces. *SIGGRAPH Comput. Graph.*, 12(3):270–274, aug 1978.