Supplementary Material for:
Data-driven Digital Lighting Design for Residential Indoor Spaces

HAOCHENG REN, HANGMING FAN, and RUI WANG, State Key Lab of CAD&CG, Zhejiang University
YUCHI HUO, Zhejiang Lab and State Key Lab of CAD&CG, Zhejiang University
RUI TANG, KooLab, Manycore Tech Inc.
LEI WANG, RaysEngine Tech Inc.
HUJUN BAO, State Key Lab of CAD&CG, Zhejiang University

1 EVALUATION OF THE PROGRESSIVE LIGHTING GUIDANCE GENERATION

To validate the progressive scheme that we use to generate lighting guidance images, we first compare the proposed approach with two image-to-image translation methods, Pix2pix [Isola et al. 2017] and SPADE [Park et al. 2019]. Table 1 shows metrics on the test set. As can be seen, both Pix2pix and SPADE are not as good as ours in all metrics. As shown in the visual comparison in Figure 1, they both exhibits some artifacts especially around lights. Our method significantly reduces the artifacts and generates more pleasing lighting effects.

We compare results that skip the step to generate a coarse image with results generated by our progressive approach. As shown in the table, generated results without the coarse image are not as good as those generated from the proposed approach. But we can see it is still better than Pix2pix and SPADE with a shading-albedo decomposition.

We also conduct experiments to evaluate image quality after the emission optimization step. Error metrics are shown in brackets in Table 1. After the optimization, all errors become lower. It demonstrates the effectiveness of our light emission optimization.

To validate the effectiveness of loss terms in the lighting guidance prediction network, we show ablation study results in Table 2.

2 DETAILS OF EXTERIOR LIGHTING

2.1 Role of Exterior Lighting

The separate interior and exterior lighting components are visualized in Figure 2. The environment light completes the relatively dark area near the door (first row), and the second row shows the sunlight predicted by our network, which boosts the realism of the lighting result.

2.2 Sunlight Prediction Pipeline

Figure 3 shows our sunlight prediction pipeline.

2.3 Lighting during Daytime or Nighttime

With the sunlight prediction pipeline, we can generate exterior lighting automatically. This is helpful for generating lighting layouts for a bunch of scenes. In some cases, the user may wish to control whether the lighting results include exterior lighting in order to consider daytime or nighttime conditions. Our system also supports this scenario. Specifically, our system does not predict the

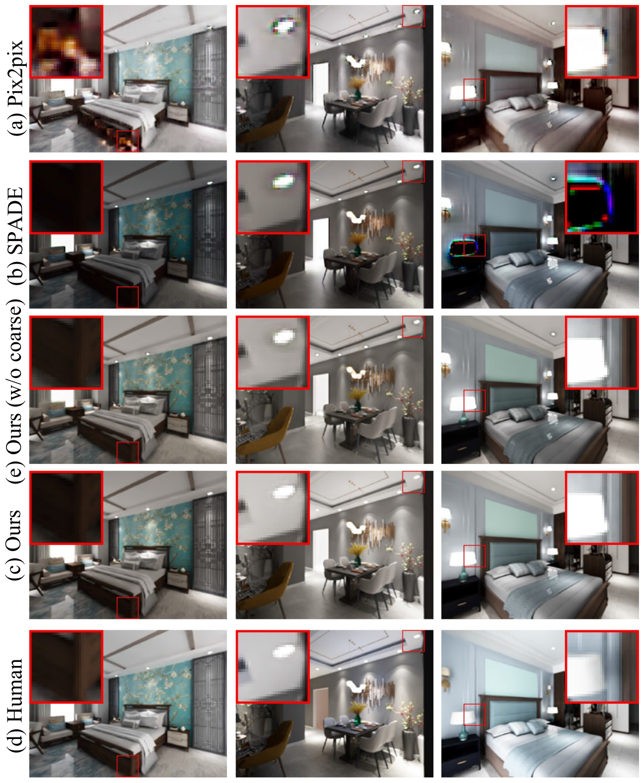


Fig. 1. Comparison of results using different lighting guidance prediction networks under the same lighting layout.

Table 1. Image Distance Metrics for Different Lighting Guidance Prediction Networks

Metrics	Pix2pix	SPADE	Ours (w/o coarse)	Ours
MSE	89.43(89.39)	104.96(97.07)	89.05(89.02)	**88.62(88.60)**
SSIM	0.85(0.85)	0.83(0.84)	0.86(0.91)	**0.89(0.91)**
PSNR	18.05(16.68)	16.86(16.71)	17.80(19.43)	**19.18(19.60)**

Results after the emission optimization step is shown in the brackets. For MSE, we calculate with HDR images which can express the lighting distribution. As SSIM and PSNR are not suitable for HDR images, LDR images are used.

0730-0301/2023/03-ART28 $15.00
https://doi.org/10.1145/3582001

Table 2. Ablation Study of the Loss Function in our Lighting Guidance Prediction Networks

Metrics	Ours (L1 only)	Ours (L1 + VGG)	Ours (L1 + GAN)	Ours
MSE	88.67	<u>88.63</u>	88.65	**88.62**
SSIM	0.881	0.883	<u>0.884</u>	**0.892**
PSNR	18.80	19.03	<u>19.05</u>	**19.18**

The best and the second best results are boldfaced and underlined.

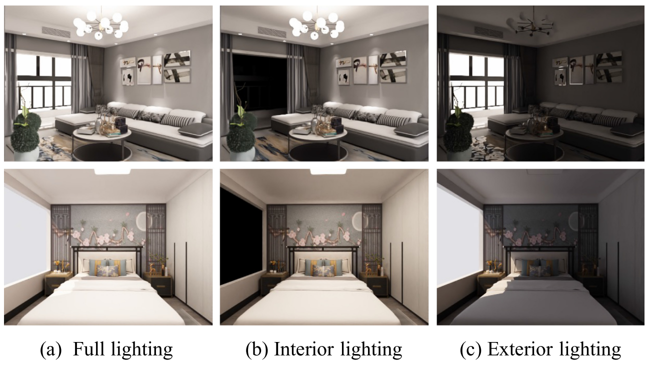


Fig. 2. Role of exterior lighting in our results. The first row shows a scene with an environment light (skylight). The second row shows a scene with exterior lighting consisting of an environment light and a sunlight. The exterior lighting is also predicted by our network and completes our interior lighting.

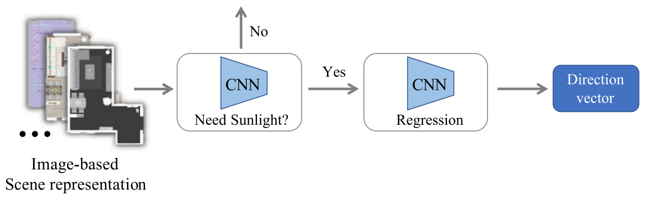


Fig. 3. The sunlight prediction pipeline. Our system can automatically generate sun light by first predict its existence, then predict its direction.

exterior lighting during the light arrangement stage in the nighttime mode, the illumination images for sunlight and environment light are black. Then, we generate guidance under nighttime with the lighting prediction network to obtain appropriate nighttime lighting automatically. Users do not need to perform the tedious process of removing the exterior lighting and tweaking the emission of every interior light manually to obtain sufficient interior lighting.

Figure 4 shows the daytime and nighttime lighting in the same room. As seen, at nighttime, the interior lighting (e.g., the lighting shed by downlights on the wall) becomes sightly brighter to compensate for the absence of exterior lighting.

3 DETAILS FOR AUTOMATIC WHOLE-ROOM LIGHTING

3.1 Evaluation of Multi-view Optimization

As we discussed in Section 6.3 in the article, optimization under one view may not be sufficient in the small fov cases, but can be

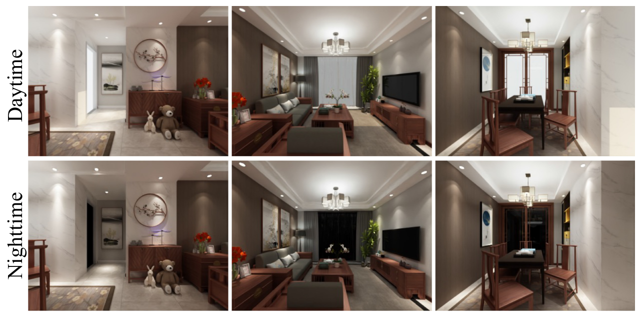


Fig. 4. Lighting design of the same room with daytime and nighttime. Users can specify the time of day in our system.

easily fixed by adding new views. We show result in these case and compare the optmization under single or mutiple views in Figure 5. The first three rows show the optimized results using the lighting guidance generated for different views. View 1 and view 2 are selected with a small field of view, while view 3 has a relatively large field of view. Different views have no or very little overlap. As shown in row 1 and row 2, lighting in uncovered areas is not guaranteed (the intensity and color may be inappropriate) if the viewpoint covers only a small portion of the room. For a view with larger coverage (row 3), the lighting results are better than for the previous views. By optimizing under two views with a small field of view (view 1 and view 2) together, the whole-room lighting is already ensured, as seen in view 3 (row 4). The last row is the lighting result optimized under all three views, and the lighting is almost the same as in row 4, which is good for the whole-room lighting. It shows that by using more views for larger scene coverage, the whole-room lighting can be obtained.

3.2 Pipeline of Whole-room Lighting Optimization

Figure 7 shows our pipeline with whole-room lighting optimization. Comparing to Figure 2 in the article, automatic camera placement module is added and light emission is optimized with multiple views.

3.3 Examples of Automatic Panorama with Optimal Polygon Partitions

We provide some examples of positions of panoramas and room partitions in Figure 6. There are on average 2.18 polygonal partitions per room in our dataset.

3.4 Calculation of Relative Pixel Footprints

To estimate the relative pixel footprints, we first record the average depth d, ray direction $\mathbf{r}$, and surface normal $\mathbf{n}$ within the pixel during ray-tracing. Then, relative pixel footprint P is calculated with the following equation. This is used as weights in the weighted non-negative least square problem.

$$P = \frac{d}{-(\mathbf{n} \cdot \mathbf{r})}$$

4 PERCEPTUAL STUDY EXAMPLE

We show our UI in the perceptual studies in Figure 8.

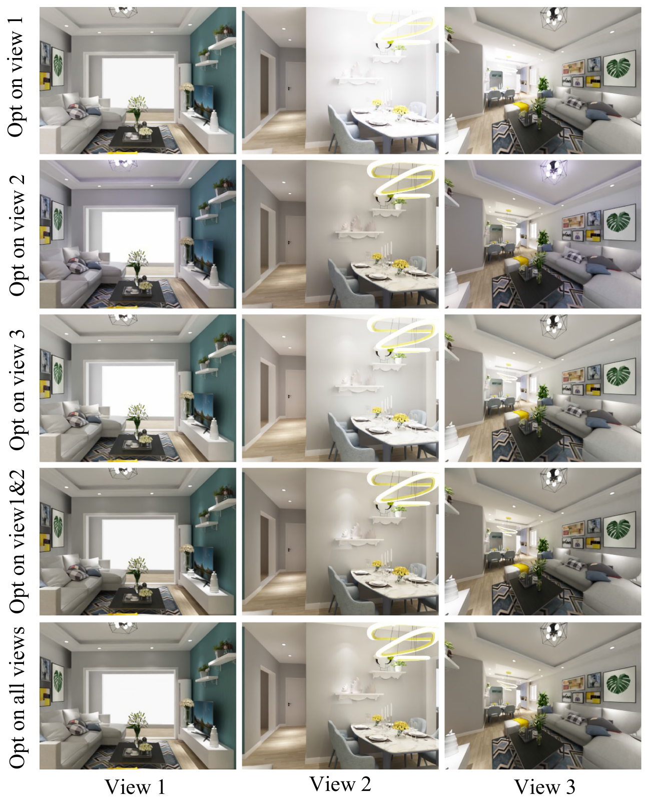


Fig. 5. Comparison of light emission optimization under one to three views. The first three rows show the optimization results under a single view, the fourth row shows the optimization under both view 1 and view 2, and the last row indicates the results optimized under all three views.

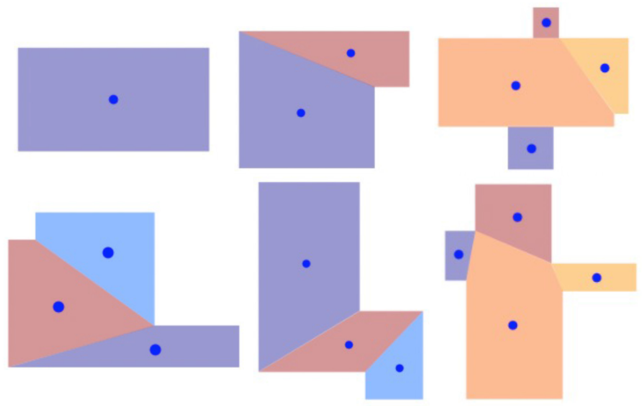


Fig. 6. Examples of optimal convex polygonal partitions of rooms in our dataset. Blue dots represent the positions of panoramas.

5 DETAILS FOR RENDERING

We use an in-house Monte Carlo path tracer for rendering all results. Classic rendering techniques like Next Event Estimation and Multiple Importance Sampling are used in the renderer. During the Next Event Estimation, we sample all light sources to reduce variance. We found it efficient especially for rendering images with each light on. We use OptiX denoiser [Chaitanya et al. 2017; OptiX 2021] for obtaining noise-free rendering images.

We use an exponential tone-mapping operator similar to that in VRay renderer [VRay 2021]. Pseudocode is provided in Algorithm 1.

ALGORITHM 1: Pseudocode of exponential tone-mapping operator.

```
dark_coef = 1.2
bright_coef = 1.5
foreach pixel c do
    brightness = $0.299 * c.r + 0.587 * c.g + 0.114 * c.b$
    $c* = max$(brightness $*$ (1.0 − dark_coef) + dark_coef, 1.0)
    $c* = min$(brightness $*$ (bright_coef − 1.0) + 1.0, bright_coef)
    $c = 1.0 - exp(-c)$
end
```

6 DETAILS OF THE RULE-BASED BASELINE

We built our rule-based baseline on the basis of an existing state-of-the-art rule-based method [Jin and Lee 2019]. We enhanced this method in several aspects to improve its quality and robustness.

Room structure loss. For the placement of the key lights (ceiling and pendant lamps), only the focal points generated by furniture groups are considered as positional targets in the original method. This approach may introduce artifacts if the room is not fully populated with furniture or is simply empty. In some cases, the focal points may also be close to the wall. Such an arrangement of key lights is usually not suitable for the structure of the room, especially when we look at the ceiling. Therefore, we add a cost term to consider the room structure. Specifically, we utilize the automatic strategy described in Section 6.3 to divide the room structure into a minimal number of convex polygonal partitions (some example partitions can be seen in Figure 6). We use the centroid of each partitioned polygon as a focal point. This strategy not only encourages the key lights to be close to the centers of regions but also encourages the lights to cover the whole room regardless of whether furniture exists in a certain region. We have also found it to be beneficial for ensuring that the key lights are not too close to the wall.

Light distribution loss. Since multiple key lights can be assigned to a single focal point in [Jin and Lee 2019], we add a cost term to encourage the lights to be distributed evenly among the focal points. To achieve this, we first count the number of lamps assigned to each focal point. Then, we calculate the variance of the numbers of lights at all focal points. The optimization process aims to minimize this variance to achieve a more uniform light distribution.

Color of lights. The original method optimizes only the light intensities. This approach simplifies the parameter space for easier optimization, but it limits the scene to be lit only by white lights. We add support for different colors of lights by optimizing both the intensity and color temperature of each light. We have also

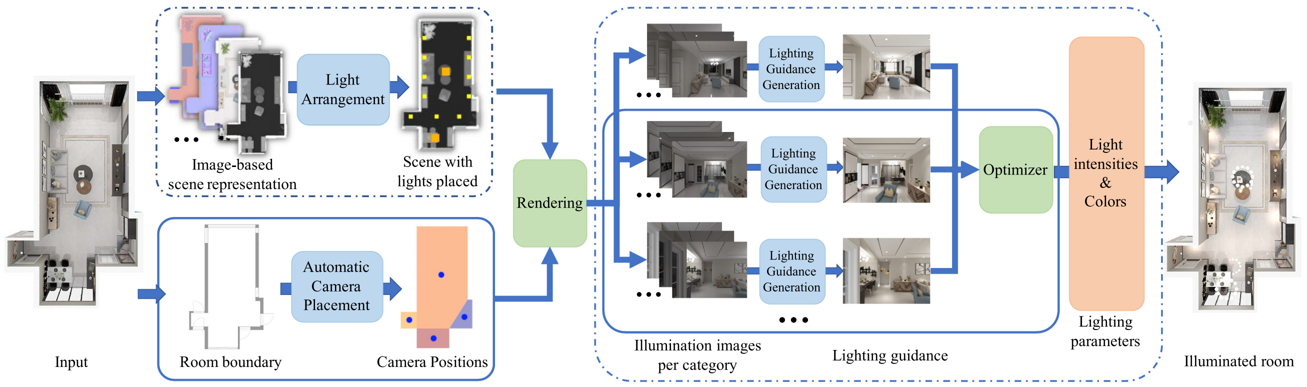
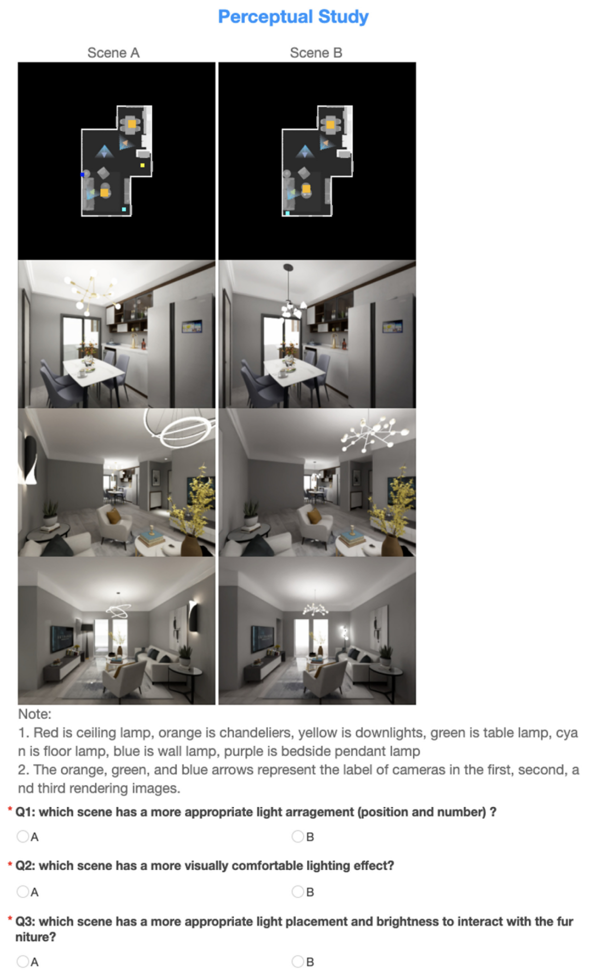


Fig. 7. Pipeline for automatic whole-room lighting design. The light emission optimization stage is extended with two modules (solid blue boxes) for generating pleasing lighting in the whole room.



Note:
1. Red is ceiling lamp, orange is chandeliers, yellow is downlights, green is table lamp, cyan is floor lamp, blue is wall lamp, purple is bedside pendant lamp
2. The orange, green, and blue arrows represent the label of cameras in the first, second, and third rendering images.

* Q1: which scene has a more appropriate light arragement (position and number) ?

○A ○B

* Q2: which scene has a more visually comfortable lighting effect?

○A ○B

* Q3: which scene has a more appropriate light placement and brightness to interact with the furniture?

○A ○B

Fig. 8. Example of user interface in our perceptual studies.

found that this approach is more robust than optimizing the RGB channels directly.

Global illumination. We use full global illumination with path tracing instead of considering only two-bounce lighting. This helps enhance the accuracy of the results to produce lighting with full global illumination, similar to our method. However, performing full path tracing during the optimization is extremely time-consuming, and it takes several hours to optimize a single scene.

Task plane of furniture. Since the original method uses the annotated surfaces of some types of furniture (e.g., the top part of a desk) as the task planes for optimizing illuminance, we extend the definition of task plane to all types of furniture in our dataset. Specifically, the task plane of each piece of furniture is the surface visible from the top-down view.

7 MORE RESULTS

Figure 9 shows more examples of designing pleasing lighting images under given camera comparing with human designers.

Figure 10 shows more examples of whole-room lighting design results comparing with human designers. We also provided videos of whole-room walk-through in the supplementary video.

8 NETWORK ARCHITECTURES AND HYPERPARAMETERS

Figures 11–18 shows the network architectures and hyperparameters of each module used in our article. Detailed description of each network's input and output is shown in Table 3. Our networks are updated with Adam optimizer [Kingma and Ba 2015].

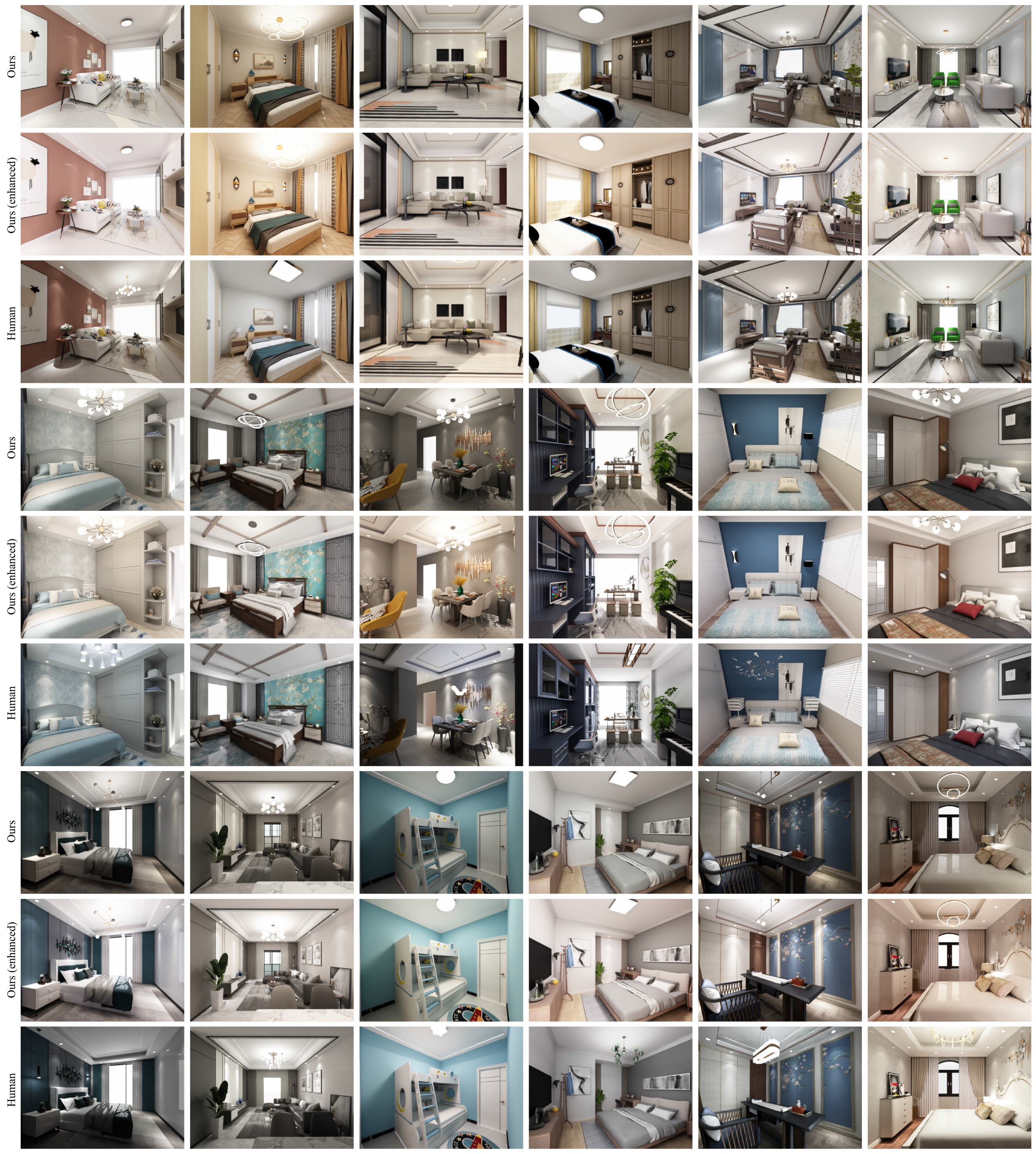


Fig. 9. Examples of our generated rendering images with pleasing lighting comparing with human designers.

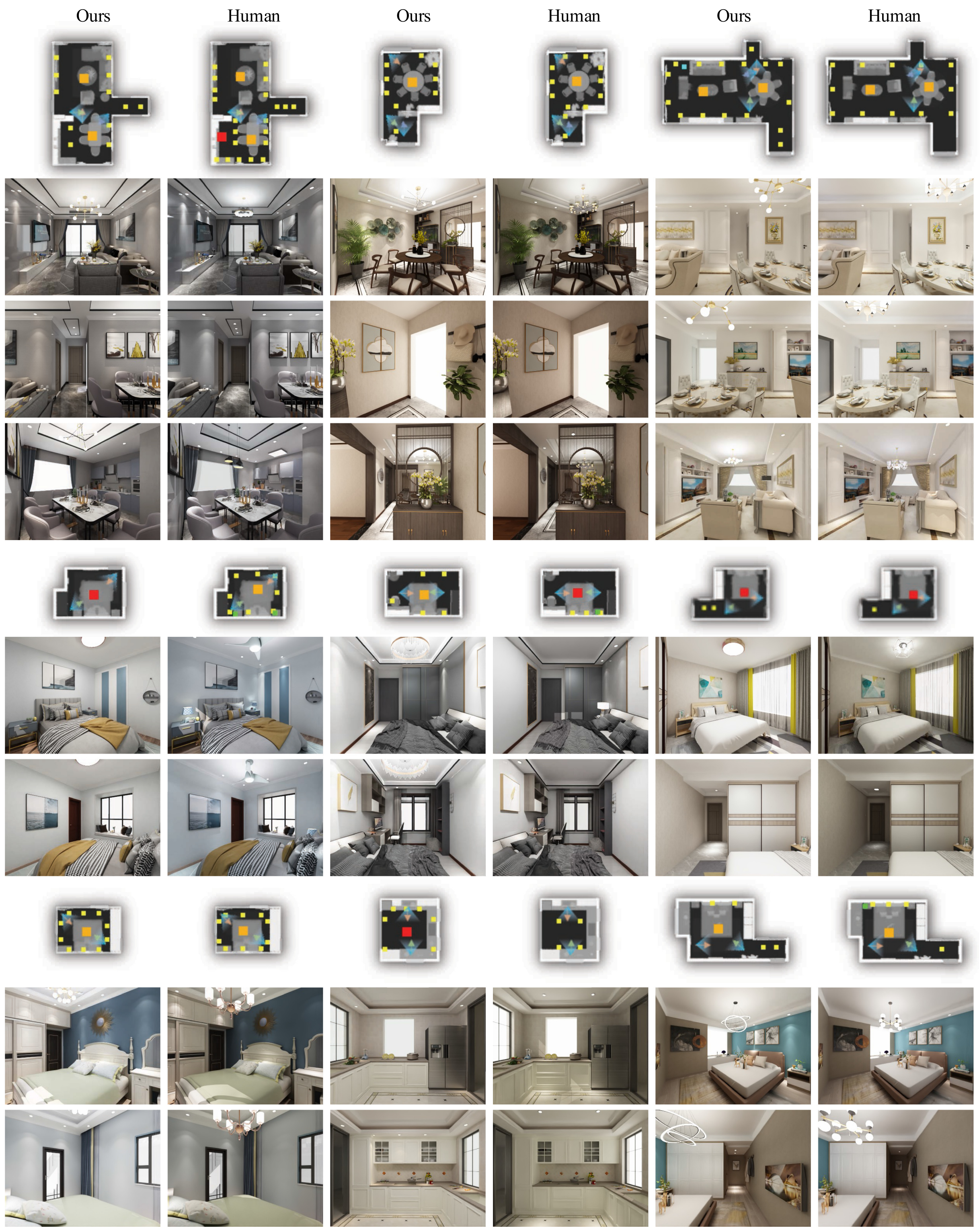
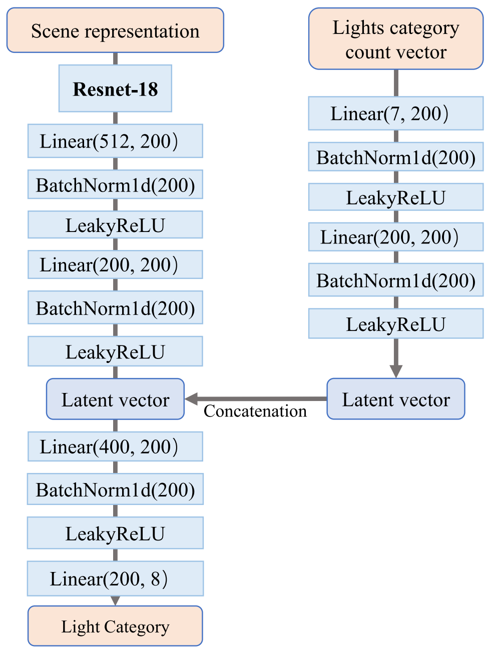
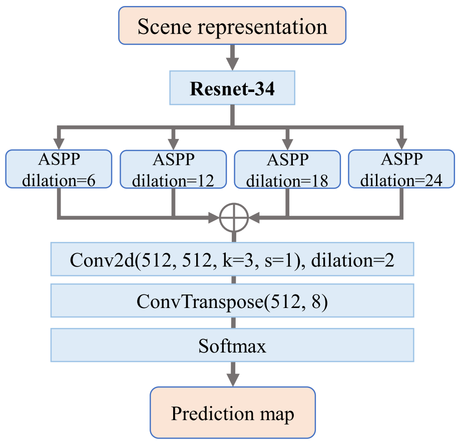


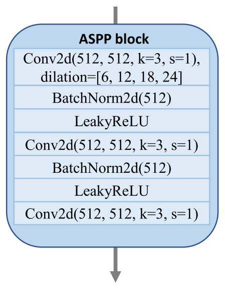
Fig. 10. Examples of our generated whole room lighting designs comparing with human designers.



Hyper Parameters	**Value**
Batch Size	16
Max Epoch	300
Max Epoch (wall)	1000
Learning Rate	1e-4
Weight decay	1e-2
Optimizer	Adam
Loss Function	Standard Cross Entropy: $L_{continuing}(x, label) = -\frac{1}{N}\sum_i^N \log(\frac{e^{x[label]}}{\sum_j e^{x[j]}})$

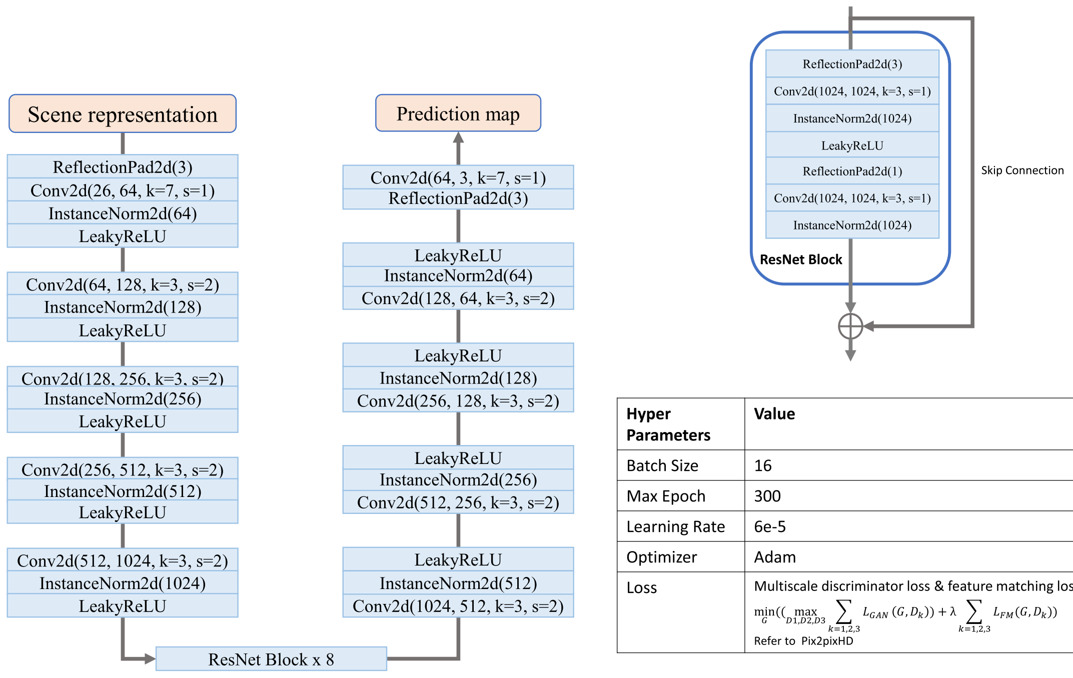
Fig. 11. NextCategoryNet.




ASPP block
Conv2d(512, 512, k=3, s=1), dilation=[6, 12, 18, 24]
BatchNorm2d(512)
LeakyReLU
Conv2d(512, 512, k=3, s=1)
BatchNorm2d(512)
LeakyReLU
Conv2d(512, 512, k=3, s=1)

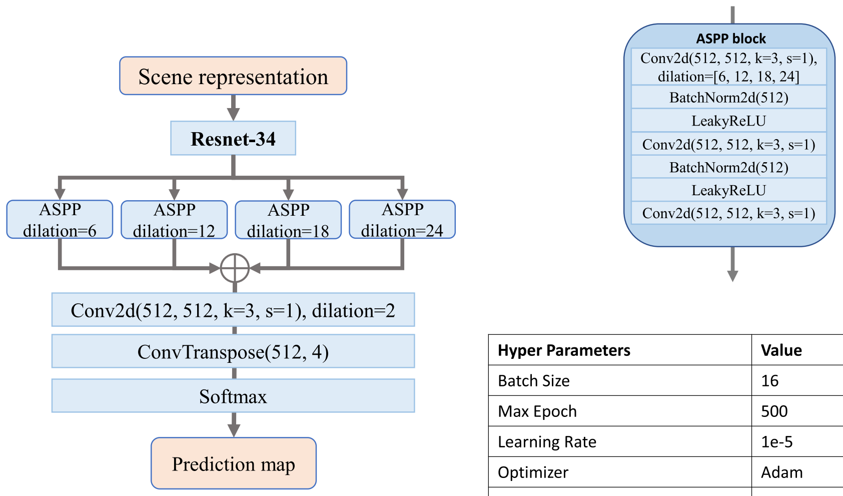
Hyper Parameters	**Value**
Batch Size	16
Max Epoch	500
Learning Rate	1e-5
Optimizer	Adam
Loss Function	Standard Cross Entropy: $L_{locating}(x, label) = -\frac{1}{N}\sum_i^N \log(\frac{e^{x[label]}}{\sum_j e^{x[j]}})$

Fig. 12. NextLocationNet.



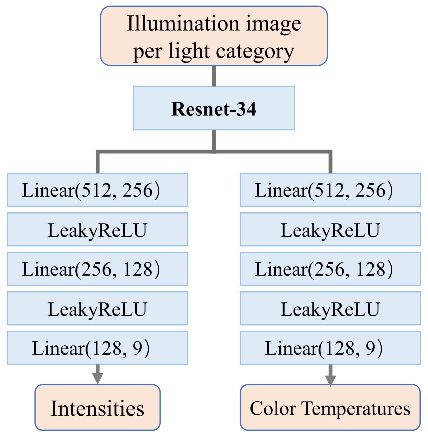
Hyper Parameters	Value
Batch Size	16
Max Epoch	300
Learning Rate	6e-5
Optimizer	Adam
Loss	Multiscale discriminator loss & feature matching loss: $\min_G((\max_{D1,D2,D3} \sum_{k=1,2,3} L_{GAN}(G, D_k)) + \lambda \sum_{k=1,2,3} L_{FM}(G, D_k))$ Refer to Pix2pixHD

Fig. 13. DownlightGAN.



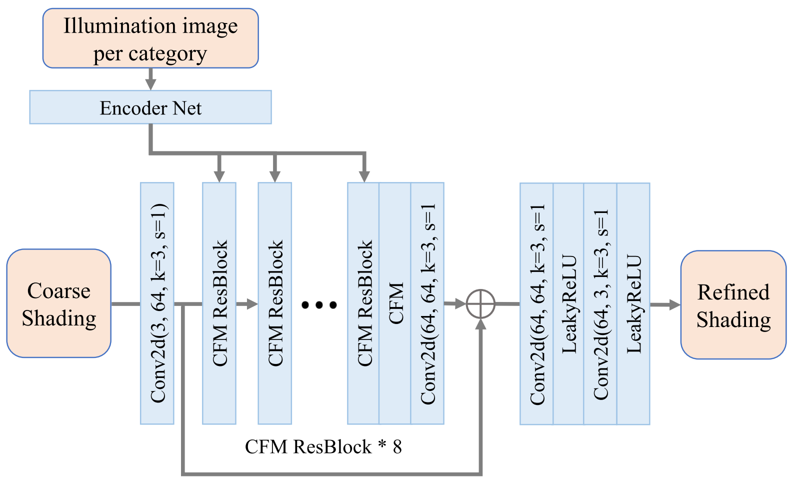
Hyper Parameters	Value
Batch Size	16
Max Epoch	500
Learning Rate	1e-5
Optimizer	Adam
Loss Function	Standard Cross Entropy: $L_{locating}(x, label) = -\frac{1}{N}\sum_{i}^{N} \log(\frac{e^{x[label]}}{\sum_j e^{x[j]}})$

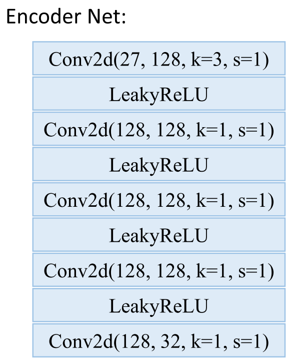
Fig. 14. WallLocationNet.



Hyper Parameters	Value
Batch Size	16
Max Epoch	200
Learning Rate	2e-5
Optimizer	Adam

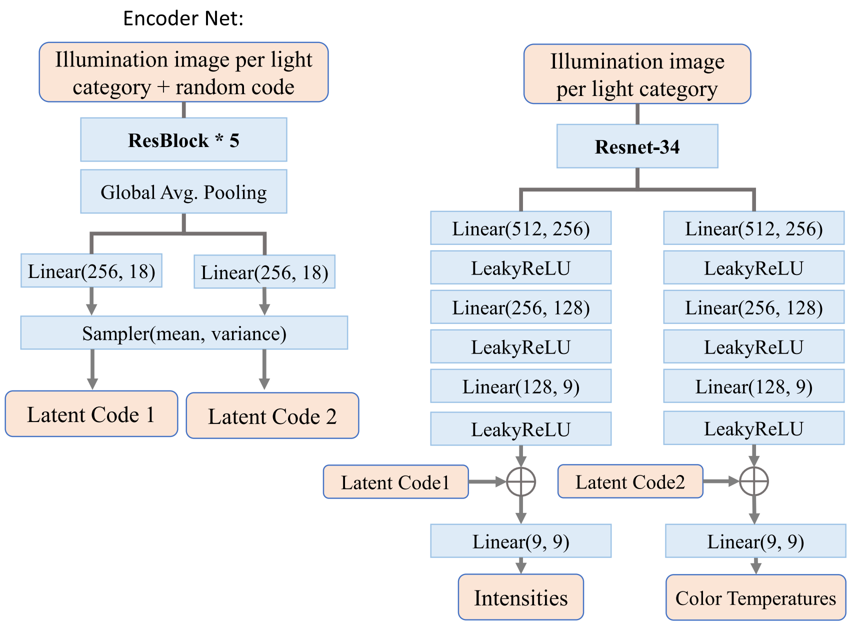
Fig. 15. IntensityNet.




Encoder Net:
Conv2d(27, 128, k=3, s=1)
LeakyReLU
Conv2d(128, 128, k=1, s=1)
LeakyReLU
Conv2d(128, 128, k=1, s=1)
LeakyReLU
Conv2d(128, 128, k=1, s=1)
LeakyReLU
Conv2d(128, 32, k=1, s=1)

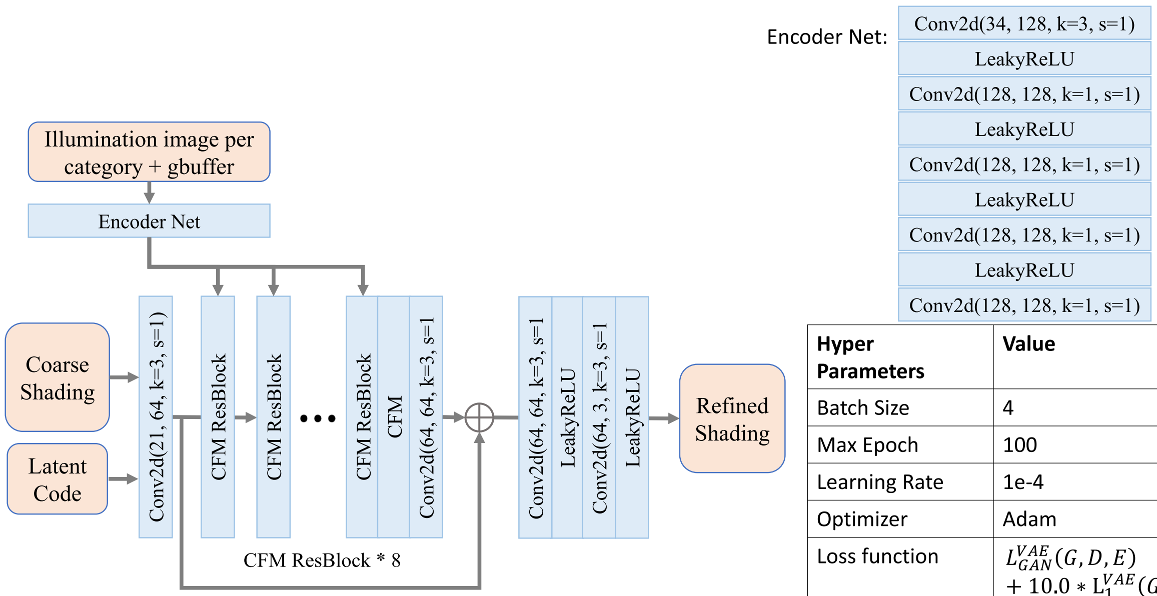
Hyper Parameters	Value
Batch Size	8
Max Epoch	200
Learning Rate	1e-4
Optimizer	Adam
Lambda VGG	0.08
Lambda GAN	5e-3
GAN Objective	LSGAN

Fig. 16. ShadingRefineNet.



Hyper Parameters	Value
Batch Size	128
Max Epoch	100
Learning Rate	2e-5
Optimizer	Adam
Loss function	$L_{GAN}^{VAE}(G,D,E) + 10.0 * L_1^{VAE}(G,E) + L_{GAN}(G,D) + 0.5 * L_1^{latent}(G,E) + 0.01 * L_{KL}(E)$

Fig. 17. IntensityNet-Multimodal.



Hyper Parameters	Value
Batch Size	4
Max Epoch	100
Learning Rate	1e-4
Optimizer	Adam
Loss function	$L_{GAN}^{VAE}(G,D,E) + 10.0 * L_1^{VAE}(G,E) + L_{GAN}(G,D) + 0.5 * L_1^{latent}(G,E) + 0.01 * L_{KL}(E)$

Fig. 18. ShadingRefineNet-Multimodal.

Table 3. The Input and Output Description of Our Networks

Network	**Input**	**Output**
NextCategoryNet	room structure(2) + top-down feature (depth(1), semantic(1), albedo(3), normal(3)) + bottom-up feature (depth(1), semantic(1), albedo(3), normal(3)) + light masks (lights for each type(7), all lights(1)) + existing lights count vector(7)	placement possibilities distribution vector(8)
NextLocationNet	room structure(2) + top-down feature (depth(1), semantic(1), albedo(3), normal(3)) + bottom-up feature (depth(1), semantic(1), albedo(3), normal(3)) + light masks (lights for each type(7), all lights(1))	light occurring probability except for the downlight(6) + INSIDE category(1) + OUTSIDE category(1)
DownlightGAN	room structure(2) + top-down feature (depth(1), semantic(1), albedo(3), normal(3)) + bottom-up feature (depth(1), semantic(1), albedo(3), normal(3)) + light masks (lights for each type(7), all lights(1))	downlight occurring probability(1) + INSIDE category (1) + OUTSIDE category(1)
WallLocationNet	room structure(2) + top-down feature (depth(1), semantic(1), albedo(3), normal(3)) + bottom-up feature (depth(1), semantic(1), albedo(3), normal(3)) + light masks (lights for each type(7), all lights(1)) + wall feature (wall top-down feature(8) + wall region mask(1) + wall light mask(1))	INSIDE category(1) + OUTSIDE category(1) + INSIDE wall but not wall lamp category(1) + WALL lamp category(1)
IntensityNet	per light category illumination(7×3) + environment light illumination(3) + sunlight illumination(3)	lighting intensities vector(9) + lighting color temperatures vector(9)
ShadingRefineNet	per light category illumination(7×3) + environment light illumination(3) + sunlight illumination(3) + coarse shading(3) + albedo(3)	refined shading(3)
IntensityNet-Multimodal	per light category illumination(7×3) + environment light illumination(3) + sunlight illumination(3) + random code (18)	lighting intensities vector(9) + lighting color temperatures vector(9)
ShadingRefineNet-Multimodal	per light category illumination(7×3) + environment light illumination(3) + sunlight illumination(3) + coarse shading(3) + albedo(3) + depth(1) + normal(3) + random code(18)	Refined shading(3)

The number in the bracket is the image channel number or the vector length.

REFERENCES

Chakravarty R. Alla Chaitanya, Anton S. Kaplanyan, Christoph Schied, Marco Salvi, Aaron Lefohn, Derek Nowrouzezahrai, and Timo Aila. 2017. Interactive reconstruction of Monte Carlo image sequences using a recurrent denoising autoencoder. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–12.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1125–1134.

Sam Jin and Sung-Hee Lee. 2019. Lighting layout optimization for 3D indoor scenes. In Computer Graphics Forum, Vol. 38. Wiley Online Library, 733–743.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*.

OptiX. 2021. OptiX. (2021). Retrieved from https://developer.nvidia.com/optix-denoiser.

Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. 2019. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2337–2346.

V. Ray. 2021. VRay. (2021). Retrieved from https://docs.chaos.com/display/VMAX/Color+Mapping.