Supplemental Document: Hierarchical Disentangled Representation Learning for Outdoor Illumination Estimation and Editing

Piaopiao Yu[1], Jie Guo[1,†], Fan Huang[1], Cheng Zhou[1], Hongwei Che[2], Xiao Ling[2], Yanwen Guo[1,†]
[1]National Key Lab for Novel Software Technology, Nanjing University, China
[2]Guangdong OPPO Mobile Telecommunications Corp Ltd, China
turmikey@163.com, {guojie, ywguo}@nju.edu.cn
{mf20330031, zc}@smail.nju.edu.cn, {chehongwei, lingxiao}@oppo.com

This document is supplemental to the paper entitled Hierarchical Disentangled Representation Learning for Outdoor Illumination Estimation and Editing. In the following sections, we provide architecture and training details for our approach. More qualitative results are also provided. All the images used in this file are from our test sets and the real world. Our networks never saw these images in the training stage.

1. Architecture and training details

1.1. Details of HDSky architecture

Our HDSky utilizes different networks to disentangle the sunny panoramas and cloudy panoramas into different lighting factors. The disentangled factors of the illumination latent space and the corresponding dimension are shown in Fig. 1.

1.1.1 Networks for sunny panoramas

As shown in the left part of Fig. 2 in the main paper, three autoencoders are used to disentangle a sunny panorama into several factors.

In AE_1, two encoders E_{sky} and E_{sun} both consist of 8 convolutional layers and a global average pooling layer [6]. The difference between the two encoders is that E_{sky} outputs a sky vector z_{sky}, while E_{sun} outputs a sun vector z_{sun}. D_{sky} is composed of 4 nearest neighbor upsampling layers with the sigmoid function as the last layer. The residual block is applied after each upsampling layer. D_{sun} first concatenates the single-channel sun position map $\mathcal{P}_{pos}$ with the feature map filled by the sun vector z_{sun}. Then, the obtained intermediate feature is mapped to the sun panorama $\mathcal{P}_{sun}$ with 3 convolutional layers and a residual block. To obtain $\mathcal{P}_{pos}$, we set the pixel values of a small rectangle (7×10) around the ground-truth sun position z_{pos} with 1. Both

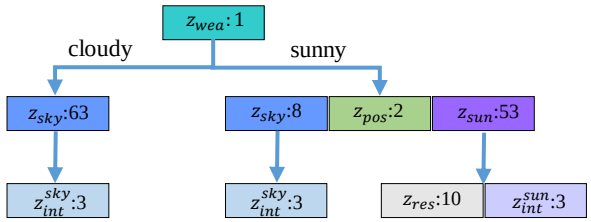

Figure 1. The hierarchically disentangled latent vectors and the corresponding dimension.

z_{pos} and the rectangle are mapped to a 32×128 panorama $\mathcal{P}_{pos}$ with latitude and longitude coordinates. Here, z_{pos} consists of the sun elevation $e \in [-\pi/4, \pi/4]$ and azimuth $a \in [-\pi, \pi]$. Then the coordinate of z_{pos} in $\mathcal{P}_{pos}$ is defined as $y = 32\times((\pi/4-e)/(\pi/2)); x = 128\times((\pi+a)/(2\times\pi))$. AE_1 is trained for 1160 epochs with a learning rate of 0.001, and then trained for 42 epochs with the learning rate of 0.00001.

The detail of AE_2 is shown in the left part of Fig. 2. F_{int}^{sun} and F_{res}, both consisting of two fully connected layers, compress a sun vector into a 3-dimensional sun intensity vector z_{int}^{sun} and a 10-dimensional residual vector z_{res}, respectively. The subnetwork D_{shp}^{sun} utilizes two upsampling layers and a softmax layer to generate the sun shape S_{sun} with 7×10 resolution. S_{sun} is then concatenated with z_{int}^{sun} and fed into the encoder E_2 which contains a convolution layer and a fully connected layer to generate the intermediate feature. Subsequently, the obtained feature is concatenated with z_{res} and fed into a fully connected layer F to achieve the reconstructed sun vector z'_{sun}.

Once AE_1 is trained, we train AE_2 with the following objective function for 600 epochs:

$$\mathcal{L}_{AE_2} = \mathcal{L}_{sun\_recon} + \mathcal{L}_{int} + \mathcal{L}_{shp} + \mathcal{L}_{res}. \quad (1)$$

Here, $\mathcal{L}_{sun\_recon}$ is the sum of the L_1 norm between (1) the input sun vector z_{sun} and the reconstructed sun vector z'_{sun}, and (2) the sun panorama $\mathcal{P}_{sun}$ and the sun panorama

†Corresponding authors.

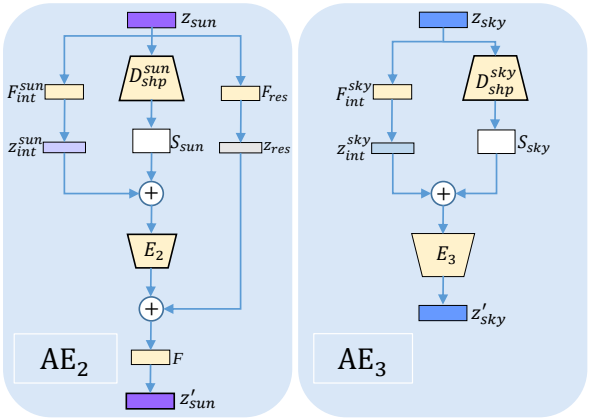

Figure 2. The details of AE_2 and AE_3 for disentanglement.

generated by z'_{sun}. We also design the intensity loss $\mathcal{L}_{int}$ to measure the similarity between the sun intensity vector z^{sun}_{int} and the one which is obtained by computing the sum of each channel of the sun area (the above 7×10 rectangle) in the sun panorama $\mathcal{P}_{sun}$.

The sun shape loss $\mathcal{L}_{shp}$ utilizes the binary cross-entropy loss to measure the similarity between the sun shape S_{sun} and the sun shape of the sun panorama $\mathcal{P}_{sun}$, which is obtained by applying a softmax operation on the sun area of $\mathcal{P}_{sun}$.

To make the residual vector z_{res} not learn the sun intensity information, we design the residual loss $\mathcal{L}_{res}$. Without supervision for z_{res}, we feed two versions of z_{res} through AE_2, one after the other: the original z_{res}, and the second one z'_{res} obtained by applying random noise on z_{res}, similar to SkyNet [2]. To enforce the sun area in the sun panoramas which are generated by the above two versions of z_{res} as close as possible, we utilize $\mathcal{L}_{res}$ to minimize the L_1 norm between the two versions of the sun area.

With the sky vector z_{sky} as input, the third autoencoder AE_3 utilizes F^{sky}_{int} which has two fully connected layers to generate the sky intensity vector z^{sky}_{int}. The subnetwork D^{sky}_{shp} utilizes 4 upsampling layers and a softmax layer to generate the sky shape S_{sky} with 32×128 resolution. Then, S_{sky} is concatenated with z^{sky}_{int} and fed into the network E_3 to generate the reconstructed sky vector z'_{sky}. E_3 is composed of 8 convolutional layers and a global average pooling layer [6]. The loss function used to train AE_3 is similar to AE_2, but does not include the residual loss. AE_3 is trained for 350 epochs.

1.1.2 Networks for cloudy panoramas

The autoencoder used to compress the cloudy panorama has an identical architecture as the sky branch of AE_1. Nevertheless, the dimension of the obtained sky vector is 63, which is slightly different from the sky vector of the sunny panorama. With the reconstruction loss (L_1 loss), the autoencoder is trained for 300 epochs with a learning rate of 0.001, and then trained for 50 epochs with a learning rate of 0.00001. 3650 HDR cloudy panoramas are used to train the autoencoder. Furthermore, we employ another autoencoder that has a similar architecture as AE_3 to disentangle the sky intensity from the 63-dimensional sky vector. The autoencoder is trained for 50 epochs with the same loss function as AE_3. 3650 63-dimensional sky vectors are used to train the autoencoder.

1.2. Details of HDSky predictor architecture

Our HDSky predictor predicts all-weather sky information from a single outdoor image with different neural networks. The classification network E_{cla}, consisting of 5 convolutional layers and a fully connected layer, outputs a single 0/1 scalar value indicating the cloudy/sunny score. E_{cla} is trained on 4,900 sunny images and 4,900 cloudy images extracted from the SUN360 dataset [7]. Convergence is obtained after 47 epochs with the learning rate of 0.001. On the test set containing 868 sunny images and 861 cloudy images, the accuracy of E_{cla} to correctly classify the weather category is 88.5%.

Our sun position prediction network E_{pos} utilizes the pre-trained DenseNet-161 [4] architecture, similar to SkyNet [2]. Predicted with E_{pos}, the elevation error of 9852 sunny images (93.7%) is less than $22.5°$, and the azimuth error of 7750 sunny images (73.7%) is less than $22.5°$ on 10,514 sunny test images.

As shown in the top-right corner of Fig. 2 in the main paper, two networks E_{sky_2} and E_{sun_2} are employed to estimate a sky vector and a sun vector of a single sunny image. The two networks both consist of 6 convolutional layers with a global average pooling layer [6] as the last layer. The residual block is applied after each convolutional layer. To estimate a sky vector from a cloudy image, we leverage a single network which is made up of 6 convolutional layers and a global average pooling layer [6].

1.3. Details of data processing

To train AE_1 in HDSky, two large datasets of HDR panoramas are employed. First, the HDR panoramas in the Laval sky dataset [5] are resized down to a resolution of 32×128 with the same approach as SkyNet [2], ensuring that the sky hemisphere remains constant. We then obtain the other dataset SUN360-HDR [2] by converting each LDR panorama of the SUN360 dataset [7] to HDR with the same approach as [2]. The HDR panoramas in the SUN360-HDR dataset are resized down to 32×128 in RGB of the up hemisphere. Then, we mask the HDR panoramas in the SUN360-HDR dataset with the sky masks which are obtained with the sky segmentation approach of [3]. In addition, we randomly apply the sky masks to the train-

Figure 3. Reconstruction quality comparison between our HDSky and SkyNet [2]. The ground-truth panoramas are from the SUN360-HDR dataset [2].

ing panoramas from the Laval sky dataset [5] with a 50% chance. AE_1 will recover the original, unoccluded sky appearance. Before fed into AE_1, all HDR panoramas are compressed by the transformation equation:

$$\tau\left(\mathrm{I}\right) = \mathrm{sign}(\mathrm{I})\frac{\log(1+\mathrm{abs}(\mathrm{I})\mu)}{\log(1+\mu)}, \tag{2}$$

where μ is set to 16 to control the amount of compression [1].

Once HDSky is trained, we run it on all panoramas in the SUN360-HDR dataset [2] to generate the corresponding disentangled vectors. These vectors together with the images extracted from the SUN360 dataset [7] are then used as training examples for HDSky predictor. The complete pipeline is shown in Fig. 2 in our main paper.

2. More qualitative results of HDSky

In this section, more visual examples from the Laval sky dataset [5] and the SUN360-HDR [2] dataset are shown in Fig. 3 to compare the reconstruction quality between our HDSky and SkyNet [2]. Compared with SkyNet [2], our HDSky achieves higher reconstruction quality with more accurate sun intensity and sky details. The reason is that our HDSky disentangles the outdoor illumination into several independent factors. The mutual interference between the sky and sun is avoided.

Figure 4. Visual comparison of outdoor illumination prediction between different methods. Our estimated lighting is more accurate than SkyNet [2] and the method of Hold-Geoffroy *et al*. [3] under different weather conditions.

3. More qualitative results of HDSky predictor

Fig. 4 shows more qualitative comparison of outdoor illumination prediction between different methods. Overall, our HDSky predictor significantly outperforms its competitors [2, 3] under different weather conditions.

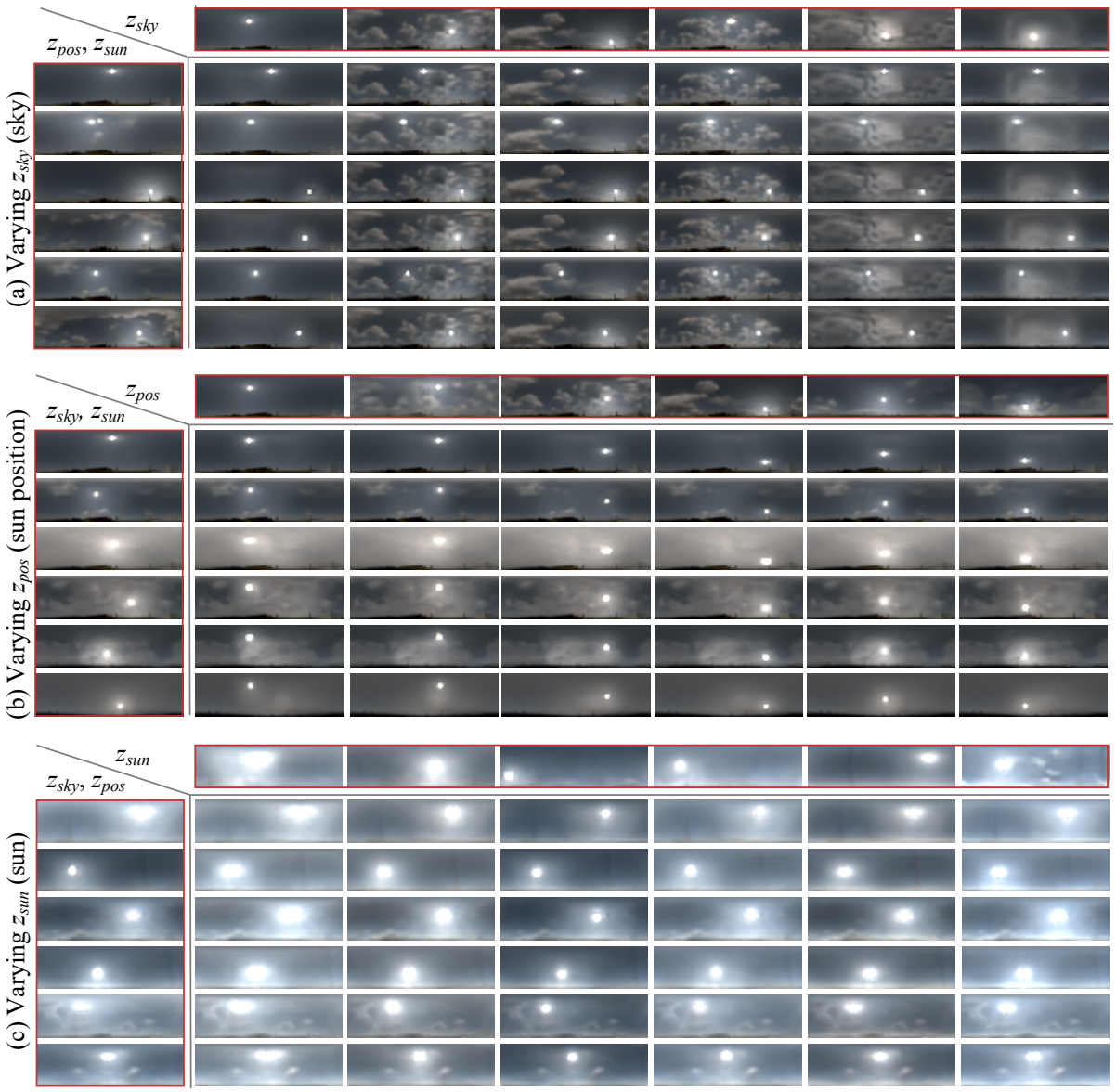

Figure 5. Varying a single lighting factor. For each subfigure, the panoramas in the top row and leftmost column (with red boxes) are reconstructed panoramas, which provide specific factors to synthesize novel panoramas in the central area. The reconstructed panoramas in the first 2 subfigures are from the Laval sky dataset [5], and the panoramas in the last subfigure come from the SUN360-HDR dataset [2].

4. More qualitative results of HDSky editor

We provide more visual results to show how well HDSky disentangles each factor and generates realistic HDR panoramas. As shown in Fig. 5, we can change the sky, the sun position and the sun of a reconstructed panorama by modifying (a) z_{sky}, (b) z_{pos} and (c) z_{sun} to synthesize novel panoramas. Fig. 5 also shows that we can change the illumination factor of a panorama by changing the specific latent vector without affecting any other illumination factors due to our hierarchical disentangled sky model.

With explicit parameters, we can directly edit the predicted outdoor illumination. Fig. 6 shows more sunny examples of intuitive edits of our HDSky editor. As seen, changing a specific lighting vector can modify the corresponding lighting factor of the predicted lighting panorama without affecting any other lighting factors. For example, by changing the sky intensity vector of the predicted panorama in the first subfigure, we can smoothly edit the sky intensity of the predicted sunny panorama and obtain the rendered images with smooth transitions of background intensity. In addition, the intensity of the sun roughly increases as the elevation angle increases, and other sky information is not affected when we edit the sun elevation.

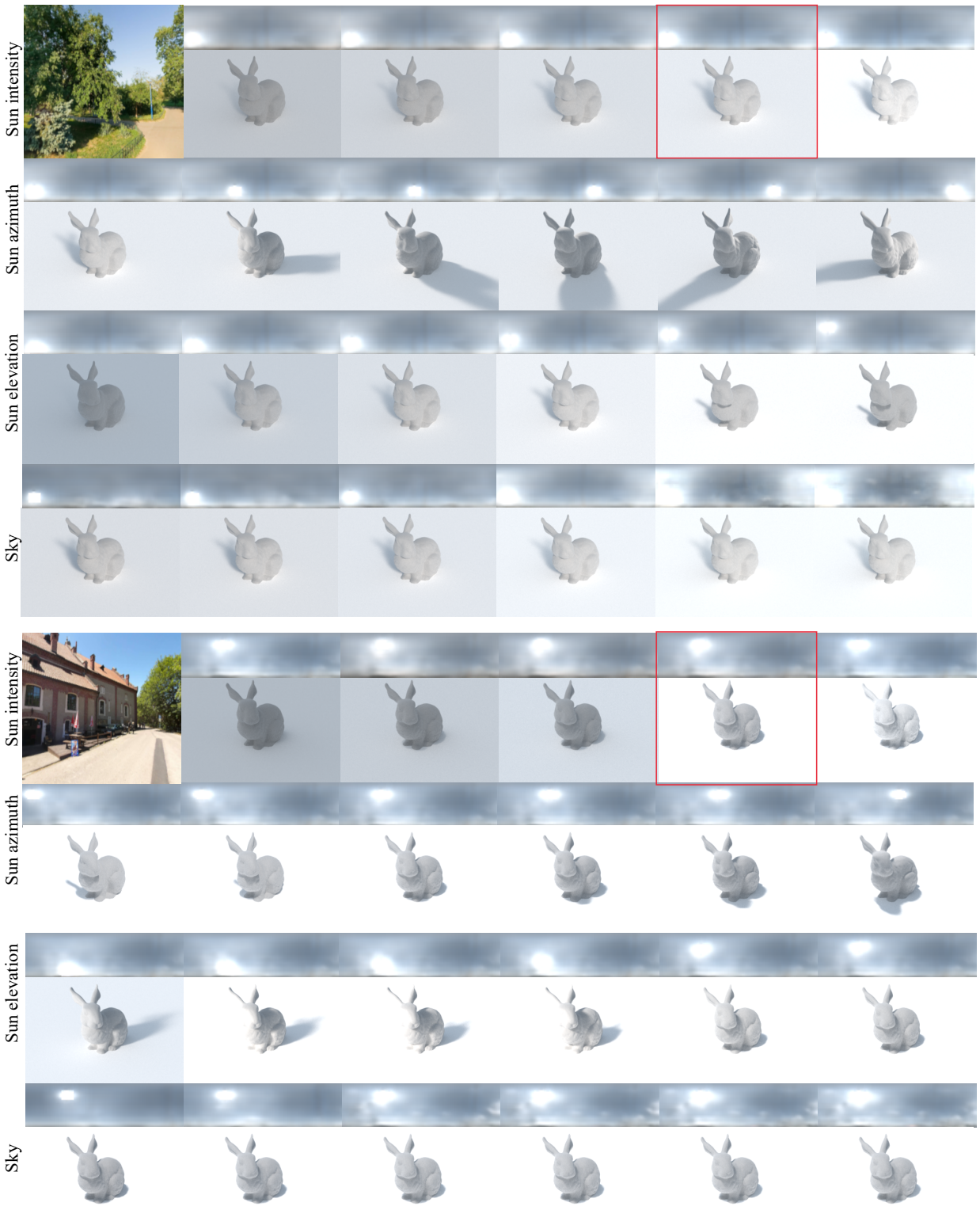

Figure 6. More sunny examples of intuitive edits of our HDSky editor. For each subfigure, the rendered image in the red box is generated with the predicted panorama of the given image in the top-left corner. Smooth transitions can be generated by changing the sun intensity, the sun azimuth, the sun elevation and the sky intensity.

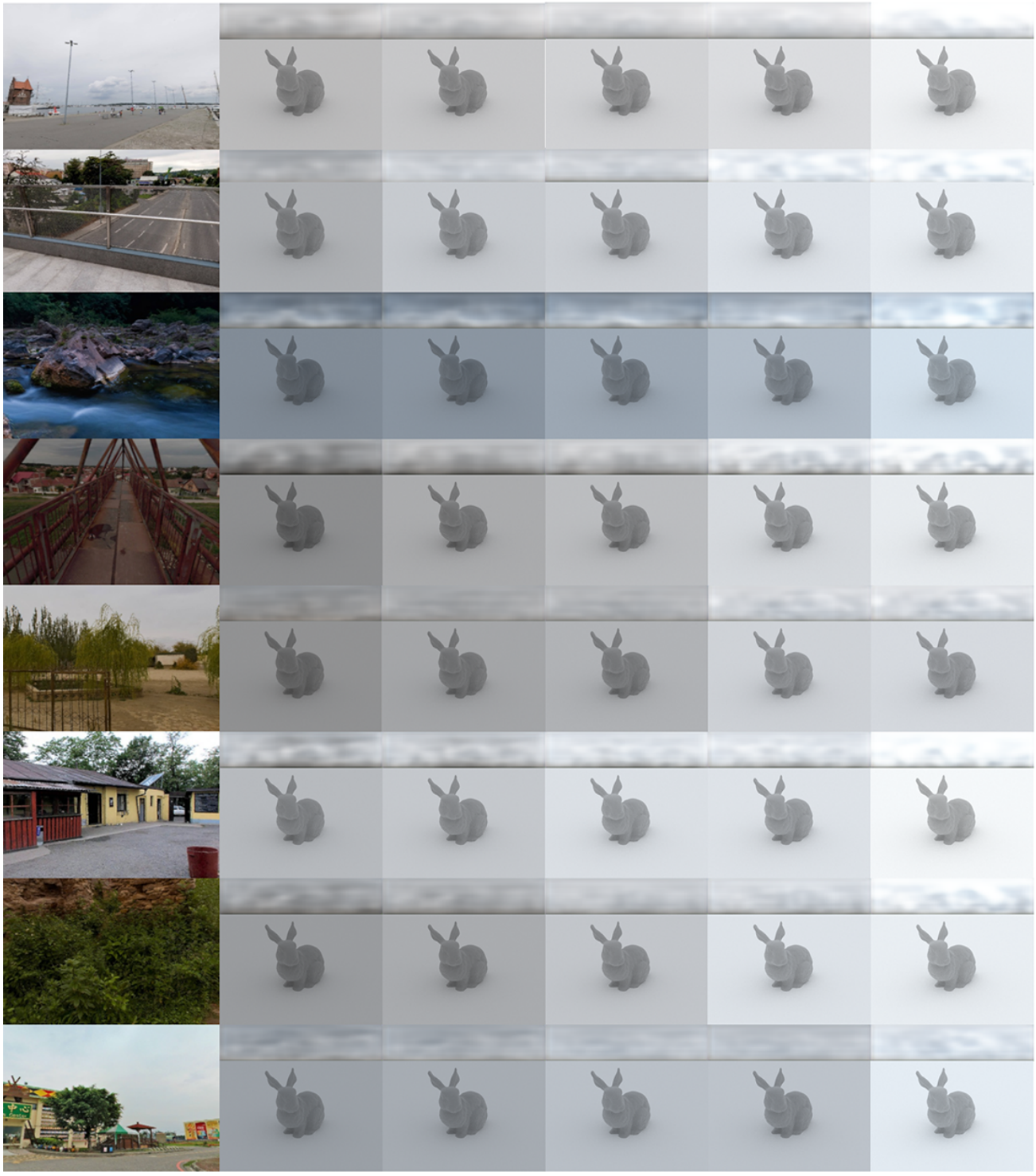

Figure 7. Intuitive edits of the sky intensity of the predicted cloudy panoramas.

For the predicted cloudy panoramas, we can edit the sky intensity with explicit parameters. Fig. 7 shows some visual examples of intuitive edits. The results indicate that changing the sky intensity vector produces smooth transitions of the sky intensity of the predicted cloudy panoramas.

Figure 8. Application of virtual object insertion of our method under different weather condition. The horse, dog, bunny and dragon are virtual objects we insert.

5. Virtual object insertion

We further show the benefit of our HDSky in the application of virtual object insertion in Fig. 8. The results reveal that our HDSky generates coherent outdoor lighting and provides plausible shadings and shadows under different weather conditions.

References

[1] Jie Guo, Mengtian Li, Quewei Li, Yuting Qiang, Bingyang Hu, Yanwen Guo, and Ling-Qi Yan. Gradnet: unsupervised deep screened poisson reconstruction for gradient-domain rendering. *ACM Trans. Graph.*, 38(6):223:1–223:13, 2019.

[2] Yannick Hold-Geoffroy, Akshaya Athawale, and Jean-

François Lalonde. Deep sky modeling for single image outdoor lighting estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 6927–6935, 2019.

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

[4] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2261–2269, 2017.

[5] Jean-François Lalonde, Louis-Philippe Asselin, Julien Becirovski, Yannick Hold-Geoffroy, Mathieu Garon, Marc-André Gardner, and Jinsong Zhang. The laval hdr sky database, 2015.

[6] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. In Yoshua Bengio and Yann LeCun, editors, *International Conference on Learning Representations*, 2014.

[7] Jianxiong Xiao, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. Recognizing scene viewpoint using panoramic place representation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2695–2702, 2012.