How to Train Neural Networks for Flare Removal: Supplement

Yicheng Wu[1] † Qiurui He[2] Tianfan Xue[2] Rahul Garg[2] Jiawen Chen[3]
Ashok Veeraraghavan[1] Jonathan T. Barron[2]
[1]Rice University [2]Google Research [3]Adobe, Inc.
wuyichengg@gmail.com jiawen@adobe.com vashok@rice.edu
{qiurui, tianfan, rahulgarg, barron}@google.com

1. Simulating random scattering flare

To simulate scattering flare with our wave optics model, we randomly sample the aperture function A, the light source's 3D location (x, y, z), and the spectral response function SRF.

Notations

- $N(\mu, \sigma^2)$: normal distribution with mean μ and standard deviation σ.
- $U(a, b)$: uniform distribution on the interval $[a, b]$.
- k_λ: the wavenumber corresponding to wavelength λ. $k_\lambda = 2\pi/\lambda$.

1.1. Aperture function

As Fig. 3(a) of the main text illustrates, we simulate dust and scratches with random dots and polylines on a clean disk-shaped aperture of radius $R = 3000$ pixels.

For each aperture function, we randomly generate $n_d \sim N(30, 5^2)$ dots with maximum radius $r_d^{\max} \sim N(100, 50^2)$. Each individual dot's radius r_d is drawn independently from $U(0, r_d^{\max})$.

Additionally, we also generate $n_p \sim N(30, 5^2)$ polylines, each of which consists $n_l \sim U(1, 16)$ line segments connected from end to end. The maximum segment length for each aperture function is $l_p^{\max} \sim N(20, 5^2)$, and each individual line segment's length is drawn independently from $U(0, l_p^{\max})$.

The opacity of each of the dots and polylines is sampled independently from $U(0, 1)$, and its location (u, v) on the aperture plane is also drawn uniformly.

1.2. Phase shift

The light source's 3D location (x, y, z) determines the phase shift ϕ_λ of the pupil function P_λ. As shown in Eq. 3 of the main text, it consists of a linear term $\phi_\lambda^{\mathrm{S}}$ and a defocus term $\phi_\lambda^{\mathrm{DF}}$.

The linear phase shift $\phi_\lambda^{\mathrm{S}}$ on the aperture plane becomes a spatial shift on the sensor plane due to the Fourier transform $\mathcal{F}\{\cdot\}$ in Eq. 4 of the main text. For an 800×800 image sensor with the center as the origin, the center of the light source (x, y) is sampled from $x, y \sim U(-500, 500)$.

The term $\phi_\lambda^{\mathrm{DF}}(z)$ is the defocus aberration due to the mismatch between the in-focus depth z_0 of the lens and the actual depth z of the light source. The analytical expression for $\phi_\lambda^{DF}(z)$ is

$$\begin{aligned}\phi_\lambda^{\mathrm{DF}}(z) &= k_\lambda \frac{u^2 + v^2}{2}\left(\frac{1}{z} - \frac{1}{z_0}\right) \\ &= k_\lambda \cdot r(u, v)^2 \cdot W_m(z)\end{aligned} \tag{1}$$

where k_λ is the wavenumber, (u, v) are aperture coordinates, and $r(u, v) = \sqrt{u^2 + v^2}/R$ is the relative displacement on the aperture plane. $W_m(z)$ is defined as

$$W_m(z) = \frac{R^2}{2}\left(\frac{1}{z} - \frac{1}{z_0}\right) \tag{2}$$

and quantifies the amount of defocus in terms of the depth z. Since we do not target any specific camera devices (i.e., specific R and z_0 values), it suffices to sample the value of W_m directly from $N(0, \sigma^2)$, with $\sigma = 5/k_{\lambda=550\mathrm{nm}}$.

1.3. Spectral response function

The spectral response $SRF_c(\lambda)$ describes the sensitivity of color channel c to wavelength λ, where $c \in \{\mathrm{R}, \mathrm{G}, \mathrm{B}\}$.

We model $SRF_c(\lambda)$ as a Gaussian probability density function $N(\mu_c, \sigma_c^2)$. The mean μ_c is the central wavelength of each channel in nanometers, and is drawn from $\mu_{\mathrm{R}} \sim U(620, 640)$, $\mu_{\mathrm{G}} \sim U(540, 560)$, and $\mu_{\mathrm{B}} \sim U(460, 480)$. The standard deviation σ_c represents the width of the passband of each color channel, and is drawn independently for each channel from $U(50, 60)$.

We then discretize the wavelength λ in $SRF_c(\lambda)$ for each of the RGB channels at 5nm intervals from $\lambda = 380$nm

†This work was done while Yicheng Wu was an intern at Google Research. He is currently a Research Scientist at Snap Research.

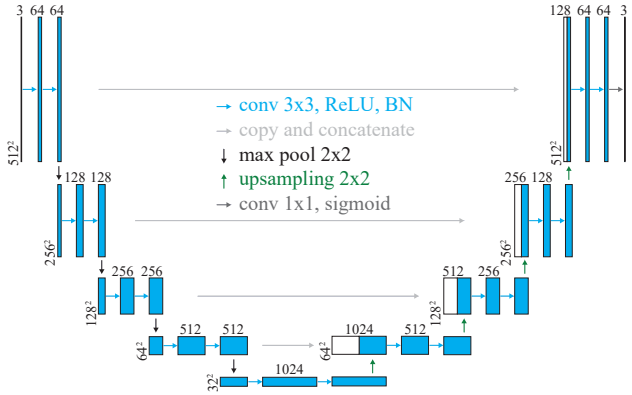


Figure 1. The U-Net architecture [6] we use in the flare removal task.

to 740nm, resulting in a 73-vector for each channel. The 3 vectors are stacked to produce an instance of SRF in the form of a 3×73 matrix.

2. Flare-only image augmentation

We augment the captured flare-only images F by applying random geometric and color transformations. For geometric augmentation, we generate a 3×3 affine transform matrix with random rotation ($\sim U(0, 2\pi)$), random translation ($\sim U(-10, 10)$ pixels), random shear ($\sim U(-\pi/9, \pi/9)$), and random scale ($\sim U(0.9, 1.2)$ for x and y independently). For color augmentation, we multiply each color channel with a random weight in $U(0, 10)$. Pixel values are clipped to $[0, 1]$ after the transformed flare image is composited with the clean image.

3. Networks and training details

To show the effectiveness of our method, we train two popular neural networks with distinct architectures. These networks have not previously been used for the task of removing lens flare. Using our method, both networks produce satisfactory results. We detail the network architectures and training configurations below.

3.1. Context aggregation network (CAN)

We repurpose a network originally designed for reflection removal [9], which is a variant of the original CAN [8]. Starting from the $512 \times 512 \times 3$ input image, the network first extracts features from a pre-trained VGG-19 network [7] at layers `conv1_2`, `conv2_2`, `conv3_2`, `conv4_2`, and `conv5_2`. Next, these features are combined with the input image to form a 1475-channel tensor, which is subsequently reduced to 64 channels with a 1×1 convolution. It is then passed through eight 3×3 convolution layers with 64 output channels at dilation rates of 1 – 64. The last layer is a 1×1 convolution with 3 output channels.

3.2. U-Net

The U-Net [6] is shown in Fig. 1. The input image I_F is $512 \times 512 \times 3$. Each convolution operator consists of a 3×3 convolution and ReLU activation. We use 2×2 max pooling for downsampling and resize–convolution for upsampling. Concatenation is applied between the encoder and decoder to avoid the vanishing gradient problem. At the final layer, we use a sigmoid function to squeeze the activations to the $[0, 1]$ range.

3.3. Training details

We train both CAN and U-Net using our semi-synthetic dataset. The clean images come from the Flickr dataset of [9]. The flare-only images are generated as described in the previous sections, and augmented on the fly during training. Training lasted 1.2M iterations (approximately 60 epochs over the 20k images in the Flickr dataset) with batch size 2 on an Nvidia V100 GPU. We use the Adam optimizer [4] with default parameters and a fixed learning rate of 10^{-4}.

4. Mask feathering

As mentioned in Sec. 5.2 of the main text, we detect the light source and create a feathered mask M_f to smoothly blend the bright illuminants into the network output.

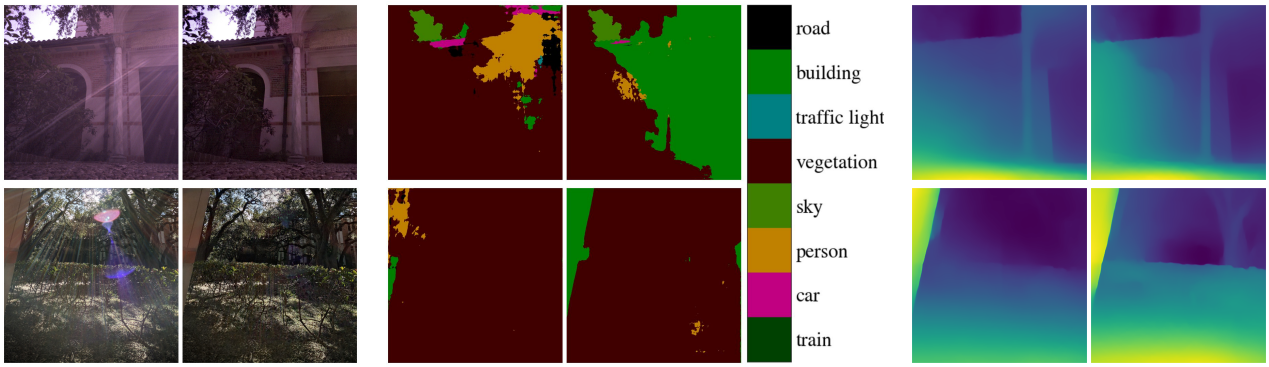
Starting from a binary mask M (the pixels where the input luminance is greater than 0.99), we apply morphological opening with a disk kernel of size equal to 0.5% of the image size. To find the primary illuminant, we partition the mask into connected components and find the equivalent diameter D of the largest region (the diameter of a circle with the same area as the region). We then blur the binary mask M using a disk kernel of diameter D. Finally, we scale the blurred mask intensity by 3 in order to guarantee that all the pixels inside the illuminant are saturated after blurring, and clip the mask to $[0, 1]$. This is the M_f used in Eq. 11 which has a feathered edge.

5. Flare removal for downstream tasks

In Fig. 2, we show that removing lens flare may benefit downstream tasks such as semantic segmentation and depth estimation. We look forward to thoroughly investigating the effect of reduced flare on a range of computer vision algorithms.

6. More results

Fig. 3 provides more visual comparisons between our method and prior work. Overall, we have 20 real test images with ground truth. The complete results, including the input images and output images from different methods, can be found at https://yichengwu.github.io/flare-removal/.



(a) Input and our output (b) Semantic segmentation [2] (c) Monocular depth [5]

Figure 2. A pair of images with flare and our output (a). Semantic segmentation (b) has much less misclassification once flare is removed using our method. Similarly, monocular depth estimation (c) is more accurate (e.g., the bush on the left in the top image and the tree in the distance in the lower image).

We also show 24 additional test images captured using the same type of lens as in Sec. 4.2 of the main text (Fig. 4), and 24 test images captured using 7 other lens types with different focal lengths (Fig. 5).

While our results are mostly satisfactory, there are certainly cases where it does not perform well. They typically have strong flare over the entire image. There is a trade-off between flare removal and scene preservation, and we prefer the latter to reduce artifacts.

References

[1] CS Asha, Sooraj Kumar Bhat, Deepa Nayak, and Chaithra Bhat. Auto removal of bright spot from images captured against flashing light source. *IEEE DISCOVER*, 2019. 4

[2] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. *ECCV*, 2018. 3

[3] Kaiming He, Jian Sun, and Xiaoou Tang. Single image haze removal using dark channel prior. *IEEE TPAMI*, 2010. 4

[4] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2017. 2

[5] Katrin Lasinger, René Ranftl, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *TPAMI*, 2020. 3

[6] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *MICCAI*, 2015. 2, 4

[7] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015. 2

[8] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *ICLR*, 2016. 2

[9] Xuaner Zhang, Ren Ng, and Qifeng Chen. Single image reflection separation with perceptual losses. *CVPR*, 2018. 2, 4

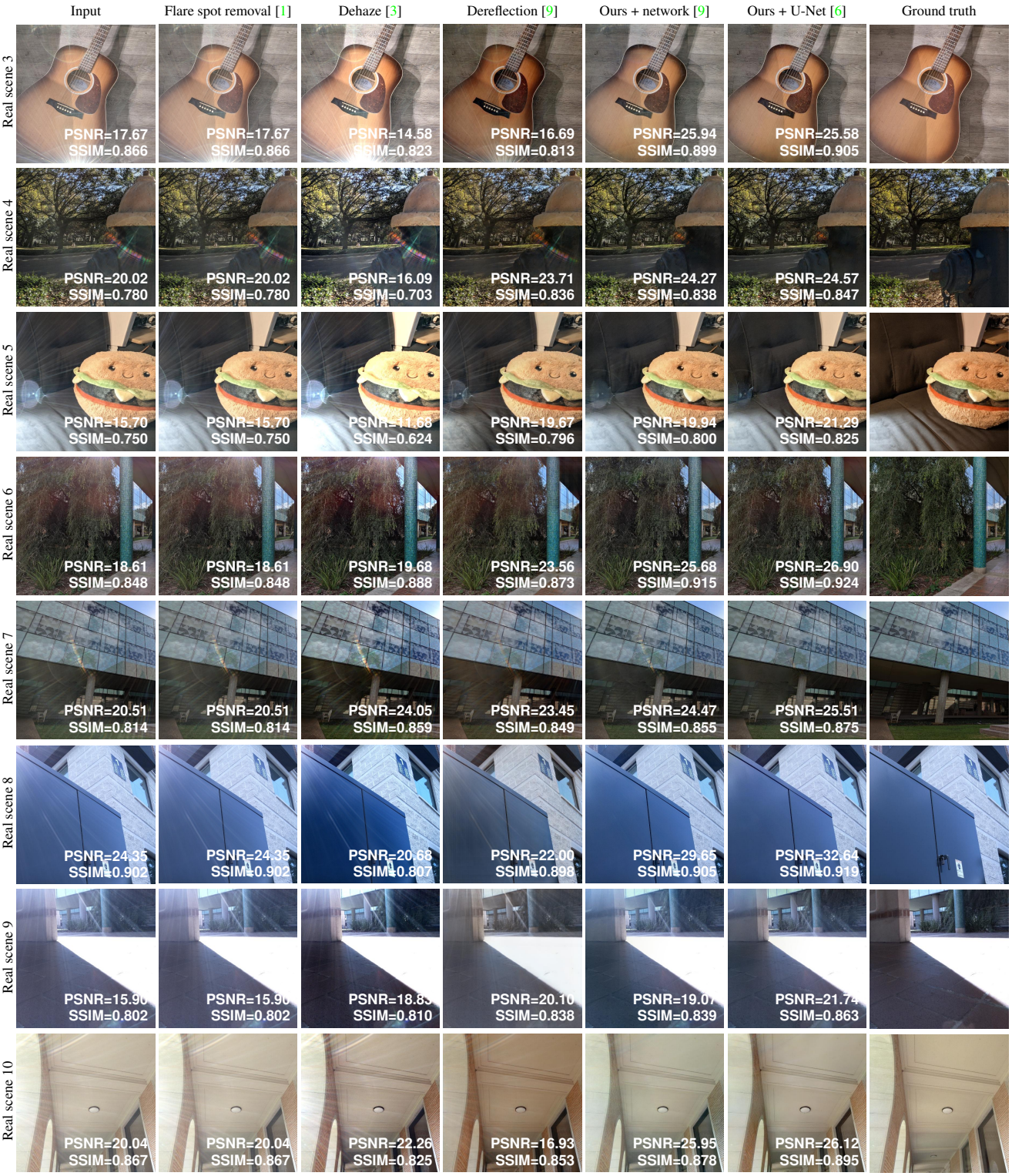


Figure 3. More visual comparison between three related methods and ours on real scenes, with PSNR and SSIM values.

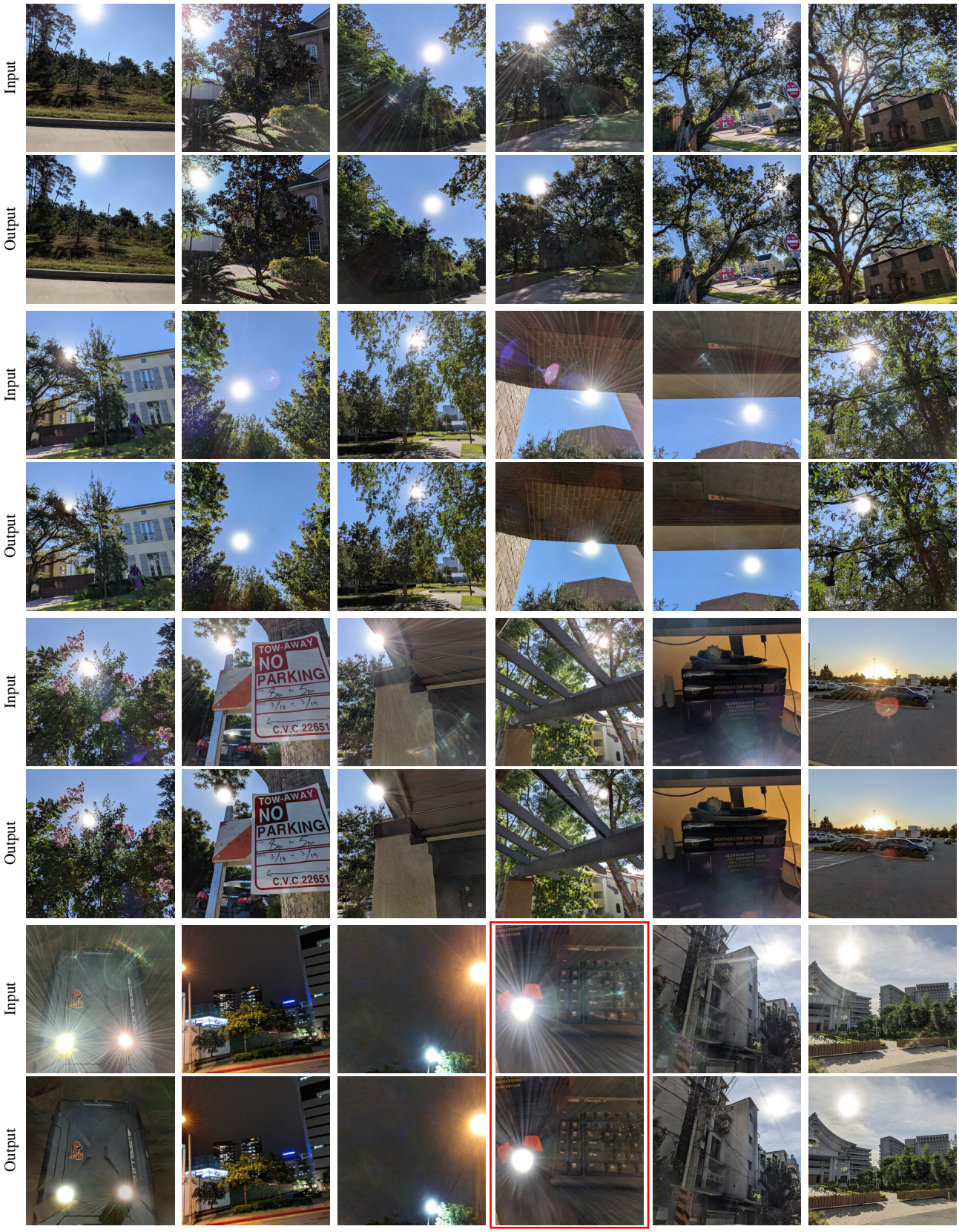


Figure 4. 24 testing images captured by the same type of lens as in Sec. 4.2 of the main text. Our method is effective in removing most of the lens flare, with occasional failures where it only removes a part of the flare (red box).

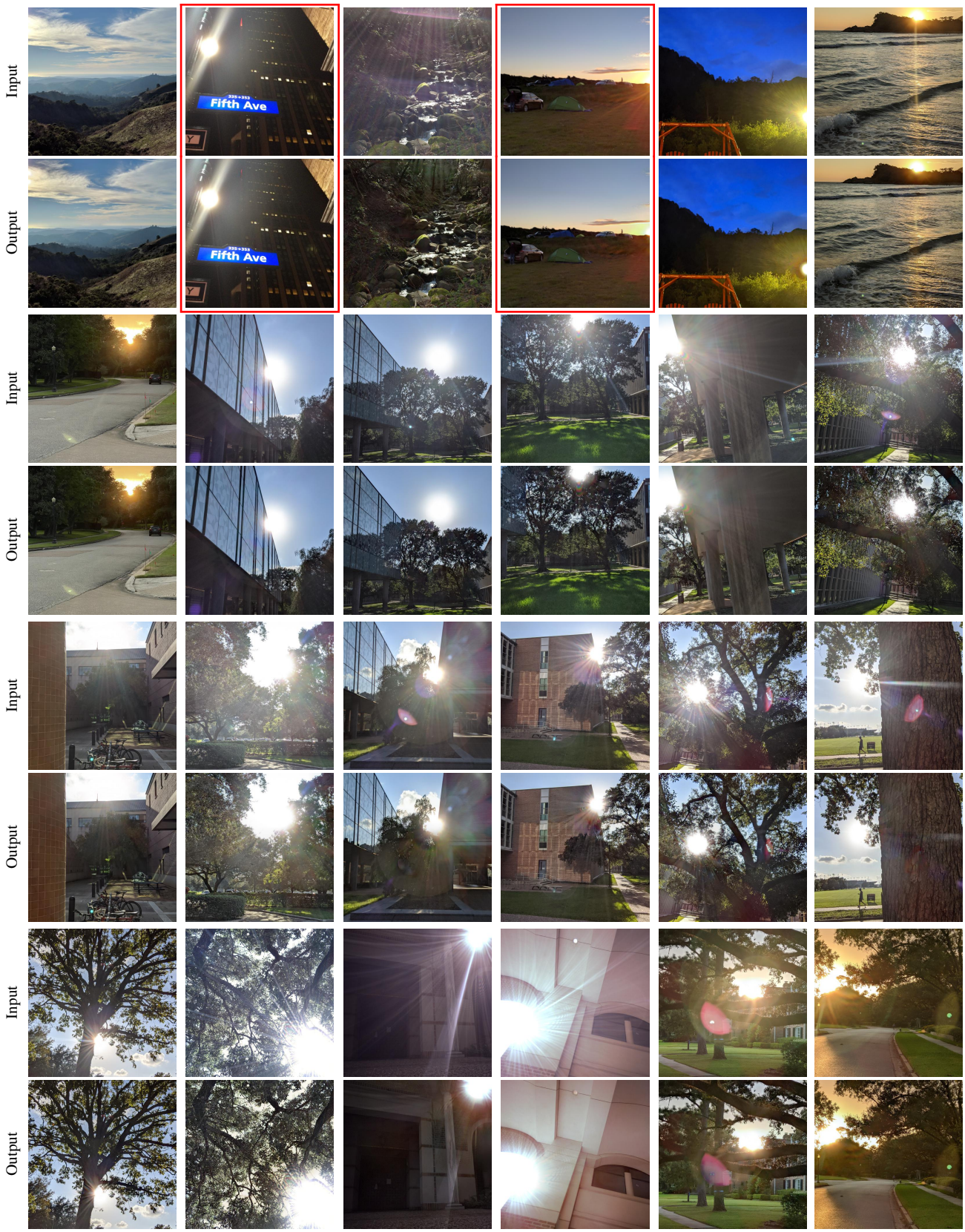


Figure 5. 24 testing images captured by 7 other lens types with different designs and focal lengths. Our method successfully remove most flares, with a few occasional failures that the algorithm either fails to identify flare (e.g., first red box) or incorrectly removes non-flare highlights (e.g., clouds in the second red box).