# An End-to-End Optimized Lensless System for Privacy-Preserving Face Verification Supplementary Material

Xin Cai[1,2], Hailong Zhang[3], Chenchen Wang[2,3], Wentao Liu[2,4], Jinwei Gu[1], Tianfan Xue[1]

[1]The Chinese University of Hong Kong, [2]Shanghai AI Laboratory, [3]Tsinghua University, [4]SenseTime
{cx023, tfxue}@ie.cuhk.edu.hk, jwgu@cuhk.edu.hk,
{zhanghl21,wcc20}@mails.tsinghua.edu.cn, liuwentao@sensetime.com

## A Simulation Pipeline from Scene to Lensless image

In Section 3 of the original paper, we illustrate the lensless imaging process by convoluting the scene's spatial intensity distribution $l_\lambda(x, y)$, with the point spread function (PSF), $p_\lambda(x, y)$, as shown in Eq.(1). In the following, we will detail the procedure for simulating the imaging process of a lensless camera with a learnable mask, using an RGB image for simulating the scene. This includes simulating the PSF with specific mask parameters, converting RGB images into spatial intensity distributions, convolving these distributions with the PSF, and finally, simulating sensor noise to generate the simulated lensless sensor images.

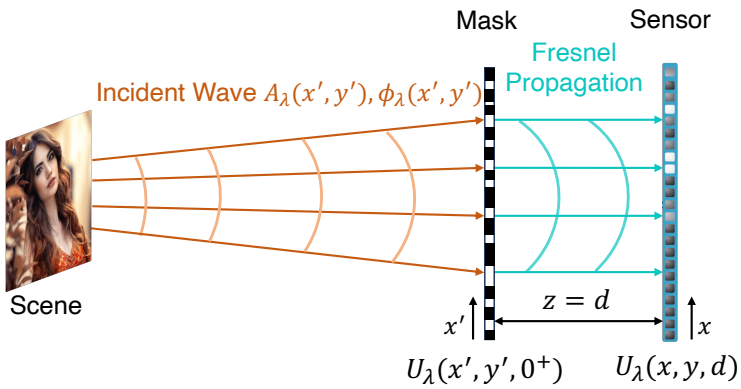


**Fig. 1:** Differentiable wave-based propagation simulation for the lensless camera. A $\lambda$-wavelength point within a scene emits an incident wave with amplitude $A_\lambda(x', y')$ and phase $\phi_\lambda(x', y')$ upon reaching the mask. The mask modulates this wave, generating a wave field $U_\lambda(x', y', 0^+)$ immediately after the mask. Following the modulation, the light propagates a distance $d$, resulting in the wave field $U_\lambda(x', y', d)$ detected at the sensor.

**Wave-based point spread function (PSF).** As illustrated in Fig. 1, consider a lensless camera consisting of a mask $M$ placed at the longitudinal position $z = 0$ and a sensor placed at $z = d$. Let $U_\lambda(x', y', z)$ be a complex scalar wave field for the wavelength $\lambda$, which is a complex function of transverse coordinates $x', y'$ and longitudinal position $z$.

The mask modulates the amplitude and the phase of the incident wave with the modulation factors $A_m(x', y')$ and $\phi_m(x', y')$ respectively. The wave field immediately after propagating through the mask is:

$$U_\lambda(x', y', 0^+) = A_\lambda(x', y')A_m(x', y')e^{i[\phi_\lambda(x',y')+\phi_m(x',y')]}, \tag{1}$$

where $A_\lambda(x', y')$ and $\phi_\lambda(x', y')$ represent the incident wave's amplitude and phase by a point source at infinity. Here we use a to-be-learned binary amplitude mask in the face verification task. We model the lensless mask $M$ with learnable parameters for transparency $w(x', y')$ at different spatial locations $(x', y')$. The modulation for the amplitude $A_m(x', y')$ attributed to the incident wave is expressed as: $A_m(x', y') = w(x', y')$. Therefore, Eq. (1) can be simplified as :

$$U_\lambda(x', y', 0^+) = A_\lambda(x', y')w(x', y')e^{i[\phi(x',y')]}, \tag{2}$$

The diffracted wave then propagates to the sensor plane at $z = d$ according to the Fresnel diffraction integral [2]:

$$U_\lambda(x, y, d) = \frac{e^{ikd}}{i\lambda d} \iint U_\lambda(x', y', 0^+)H(x, y; x', y')dx'dy', \tag{3}$$

where wave number $k = 2\pi/\lambda$ and the free-space propagation frequency response $H(x, y; x', y') = e^{\frac{ik}{2d}[(x-x')^2+(y-y')^2]}$ . The point spread function (PSF) corresponding to the point source at infinity is:

$$p_\lambda(x, y) = |U_\lambda(x, y, d)|^2. \tag{4}$$

The process of light modulation via a mask to generate a PSF is fully differentiable. We can optimize the mask parameters $\theta_M = \{w(x', y')\}$ to obtain the desired light modulations and PSF, thereby improving the performance of downstream tasks. Note that in amplitude masks, the PSF $p_\lambda(x, y)$ remains consistent across any wavelength $\lambda$ and we can drop the wavelength parameter.

**Convert RGB images to spatial intensity of scenes.** To clearly define this imaging simulation challenge, consider an RGB image $I \in \mathbb{R}^{C\times H\times W}$, where $C$ denotes the number of color channels. Accompanying this, we have a PSF $P \in \mathbb{R}^{C\times H_P\times W_P}$. The objective is to produce an image $M \in \mathbb{R}^{C\times H_P\times W_P}$ that represents what a lensless sensor would capture.

First, we convert the RGB image $I$ into a scene light spatial intensity map $l(x, y) \in \mathbb{R}^{C\times H_{sc}\times W_{sc}}$. This transformation enables us to simulate the effects of a specific physical imaging setup by convolving the intensity map with the digital PSF $P$. The imaging setup is characterized by the following parameters: the scene's height $h$, the distance from the scene to the encoder $d_1$, and the

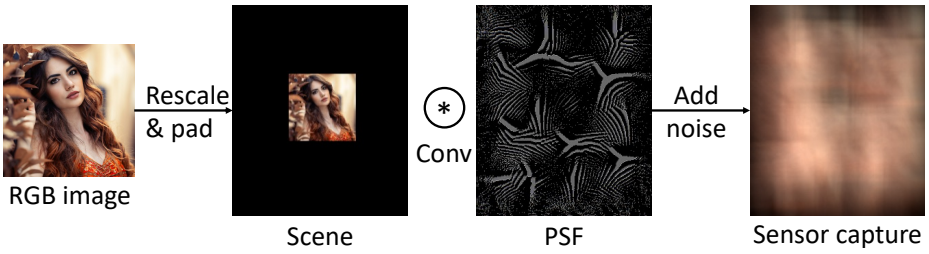


**Fig. 2:** Simulation from the RGB image to sensor capture.

distance from the encoder to the image plane $d_2$. For such a configuration, the scene height at the sensor is given by:

$$\hat{h} = \frac{d_2}{d_1} h. \tag{5}$$

Now, if we have a PSF that is captured at the given distances with a resolution of $H_P \times W_P$ pixels and a pixel pitch of $p$, the spatial resolution on the sensor corresponding to the scene's height $\hat{h}$ is calculated in pixels as:

$$H_{sc} = \frac{\hat{h}}{p}. \tag{6}$$

Next, we resize the RGB image $I$ to match the spatial resolution of $\hat{h}$ on the sensor, resulting in $l(x, y) \in \mathbb{R}^{C \times \text{round}(H_{sc}) \times \text{round}(W_{sc})}$, where the scaling factor $S$ is computed using $S = (H_{sc}/H)$. This means we scale the height $H$ and width $W$ of the original image by $S$ to match the sensor's calculated spatial resolution $H_{sc}$. Finally, to facilitate proper convolution with the PSF, we pad $l(x, y)$ with zeros to match the PSF's dimensions, resulting in a final intensity map $\hat{I}$ of shape $\mathbb{R}^{C \times H_P \times W_P}$. This ensures compatibility between the intensity distribution and the PSF in the subsequent convolution operation required for simulating the lensless camera's image capture. Fig. 2 schematically illustrates the entire simulation procedure.

**Convolve the rescaled scene with PSF.** After rescaling the image to a final intensity map with the same shape as the PSF, we then can do the convolution between the rescaled scene and PSF, to obtain the convolution result $S \in \mathbb{R}^{C \times H_P \times W_P}$:

$$S = \hat{I} * P. \tag{7}$$

It is noted that the convolution kernels $P$ have a large shape, making it more efficient to perform convolution in the spatial frequency domain rather than the spatial domain. In the spatial frequency domain, convolution becomes an element-wise multiplication, allowing for the use of the Fast Fourier Transform (FFT) algorithm to efficiently switch between the spatial and spatial frequency domains.

**Add Noise on Sensor.** The convolution result $R$ is integrated over the sensor pixels and corrupted by read and shot noise [3], yielding a measurement $M$ on the sensor given by:

$$M \sim \mathcal{N}(\mu = S, \sigma^2 = \lambda_{read} + \lambda_{shot} S). \tag{8}$$

where $\lambda_{read}$ and $\lambda_{shot}$ represent coefficients of read and shot noise respectively and are determined by the sensor's analog and digital gains [1], $\mathcal{N}(\mu, \sigma^2)$ is the normal distribution with mean $\mu$ and variance $\sigma^2$.

## B Simulation of Varying Lighting Conditions

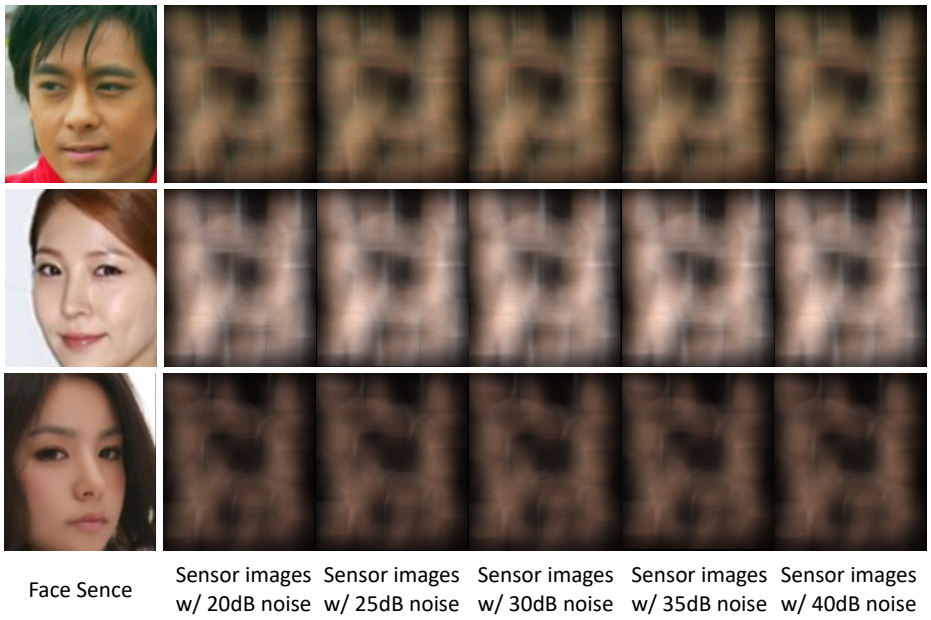


**Fig. 3:** Simulation from the RGB images to sensor captures with varying levels of noise.

In Section 5.7 of the original paper, we explore the simulation of varying lighting conditions for the lensless camera by adding different levels of noise. By carefully manipulating the sensor's digital gains, we can precisely control the noise level on the lensless sensor capture, achieving specific noise levels, such as an SNR of 20 dB, as shown in Eq. (8). Additionally, Fig. 3 present several examples of lensless images that have been modified with varying degrees of noise, illustrating the effects of these conditions on image quality.

## C Optimization of Binary Amplitude Mask

To optimize a binary amplitude mask during end-to-end training, we utilize a custom differentiable "binarizing" function. During the forward pass, the function takes the real-valued mask weights $w$ and converts them to binary by applying a round operation. However, this round operation is not backward compatible, so we cannot propagate gradients through it. To address this, we first clip the weights between 0-1, then calculate the binary version by rounding. We take the detached version of this as the final output. However, for the backward pass, we want the gradients to flow through the original weights $w$ before clipping. Therefore, we subtract the detached clipped weights from the binary output and add back the original clipped weights. This acts as an identity operation for the backward pass while still providing a binary output. This allows us to optimize the real-valued $w$ during backpropagation to learn the optimal binary mask. Here is the pseudo-code of the "binarizing" function and Fig. 4 shows the gradient of the binarizing function:

```
def Binarize(w):
    clipped_weights = clamp(w, min=0, max=1)
    binary_weights_no_grad = round(w)
    binary_weights =
        binary_weights_no_grad.detach()
        - clipped_weights.detach()
        + clipped_weights
    return binary_weights
```

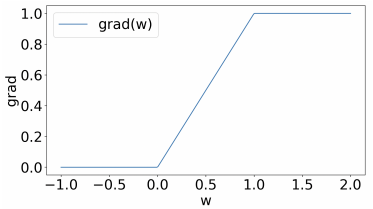


**Fig. 4:** The gradient of binarizing function.

## D Implementation Details of Reconstruction Attacks

In Section 5.3 of the original paper, we introduce to simulate the reconstruction attacks with the RGB-lensless face pair data. We trained a U-Net [4] as a decoder for simulating reconstruction attacks. Here we introduce the implementation details of the training. The lensless captures, originally sized at 240x200 pixels, were first padded to achieve a resolution of 256x256 pixels. These padded images were then fed into a U-Net model characterized by a feature dimension of 64, with the smallest resolution of the feature maps being 16x16 pixels. To accommodate the varying scales of the attacks simulated—ranging from 10 plaintext attacks to 10,000—we adjusted the batch sizes accordingly: 2 for the 10 plaintext attacks scenario, 4 for the 100 attacks, 16 for the 1,000 attacks, and 32 for the 10,000 attacks scenario. All models were trained using the Adam optimizer, with a learning rate set at 2e-4, across a total of 50 epochs.

## References

1. Brooks, T., Mildenhall, B., Xue, T., Chen, J., Sharlet, D., Barron, J.T.: Unprocessing images for learned raw denoising. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11036–11045 (2019) 4

2. Goodman, J.W.: Introduction to Fourier optics. Roberts and Company publishers (2005) 2
3. Hasinoff, S.W.: Photon, poisson noise. Computer Vision, A Reference Guide **4**(16), 1 (2014) 4
4. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015. pp. 234–241. Springer (2015) 5