4. Methods

We show in this section how we compute the phase-only hologram H which we display on SLM, how we encode the amplitude in phase-only terms, and how we simulate the reconstructed hologram views via backward propagation. In our implementation, we performed both forward and backward propagation between the SLM plane and the hologram plane, the former for computing the hologram H, and the latter for visualizing a simulated reconstruction, which we showed later in the results section.

4.1. Generation of hologram: forward propagation

We compute the hologram using the Fresnel holography principle adopted by Maimone[1]. A beam is focused at finite distances by a hologram, to several object points, which are formed by overlapping regions that are sub-holograms. Each sub-hologram is a phase-altering function like a refractive lens. The lens phase function, or sub-hologram, $f_{\text{forward}}(\vec{p})$, for each pixel, is defined as:

$$f_{\text{forward}}(\vec{p}) = e^{j(\phi_0 + \frac{2\pi\|\vec{p}-\vec{o}\|}{\lambda})}$$

where $\vec{p}$ is the location on the SLM and $\vec{o}$ is the location of the projected hologram in space. ϕ_0 is initialized to be a random phase for each pixel.

Then the hologram is a summation of product the amplitude at each pixel with the lense phase function at that pixel:

$$H(\vec{p}) = \sum_{j \in s_p} a_j f(\vec{p}_j))$$

This is essentially a convolution. In our implementation, for single-depth images, we computed this as the Inverse Fourier Transform of the product of the Fourier Transforms of $f_{\text{forward}}(\vec{p})$ and I where I is the input intensity image. We name this forward propagation, from the SLM plane to the hologram volume.

For multi-depth images, for depth $d \in [R]$, we perform the above propagation for each I_d, which is the input intensity image having only intensities at pixels of depth d and zero elsewhere.

$$H_d = IFFT\Big(FFT\left(f_{\text{forward}}(\vec{p})\right) \cdot FFT\left(I_d\right)\Big)$$

Then after obtaining the corresponding hologram H_d, we sum all H_d for all depths and the summation

$$H = \sum_{d \in \mathbb{R}} H_d$$

which is our generated multi-depth hologram. Per-pixel focal control is thus allowed as different pixels can be focused at different depths.

4.2. Simulating the reconstructed hologram views: backward propagation

To simulate the reconstructed hologram view at depth d, we perform backward propagation, by taking

$$I_d = IFFT\Big(FFT\left(f_{\text{backward}}(\vec{p})\right) \cdot FFT\left(H\right)\Big)$$

where

$$f_{\text{backward}}(\vec{p}) = e^{-j(\phi_0 + \frac{2\pi\|\vec{p}-\vec{o}\|}{\lambda})}$$

Then we have I_d is the simulated propagated hologram view at depth d.

4.3. Encoding the Amplitude

Double phase encoding. Since we are working with a phase-only SLM, we need a method to encode the amplitude in the phase. We adopt the same method used by [1]. We can encode any complex value c with amplitude $a \in [0, 1]$ (our input intensity image is normalized) and phase p into the sum of two values $c = ca + cb$:

$$c = ae^{ip}$$

$$p_a = p - \cos^{-1} a, p_b = p + \cos^{-1} a$$

$$c_a = 0.5e^{ip_a}, c_b = 0.5e^{ip_b}$$

Thus we convert the complex-valued hologram H into summation of the following phase-only holograms:

$$P_a(\vec{p}) = \angle H(\vec{p}) - \cos^{-1}|H(\vec{p})|$$

$$P_b(\vec{p}) = \angle H(\vec{p}) + \cos^{-1}|H(\vec{p})|$$

Note that we require two phase values and thus two pixels for each complex value in H. We expect to see loss in the resolution.

Direct removal of intensity. Alternatively, we also experimented with directly removing the amplitude of the complex-valued hologram $H = Ae^{j\theta}$ treating $A = 1$ across the image and encoding the angle directly using euler's euqation. The phase-only hologram HP is computed as:

$$HP = \cos(\theta) + j * \sin(\theta)$$

We observe a significant decrease in the noise using double phase method than direct removal of intensity.