Supplement
Michelson Holography: Dual-SLM Holography with Camera-in-the-loop Optimization

1. IMPLEMENTATIONS

A. Hardware Implementation

In addition to the hardware details in the main paper, we show in Fig. S1 our bench-top display prototype.

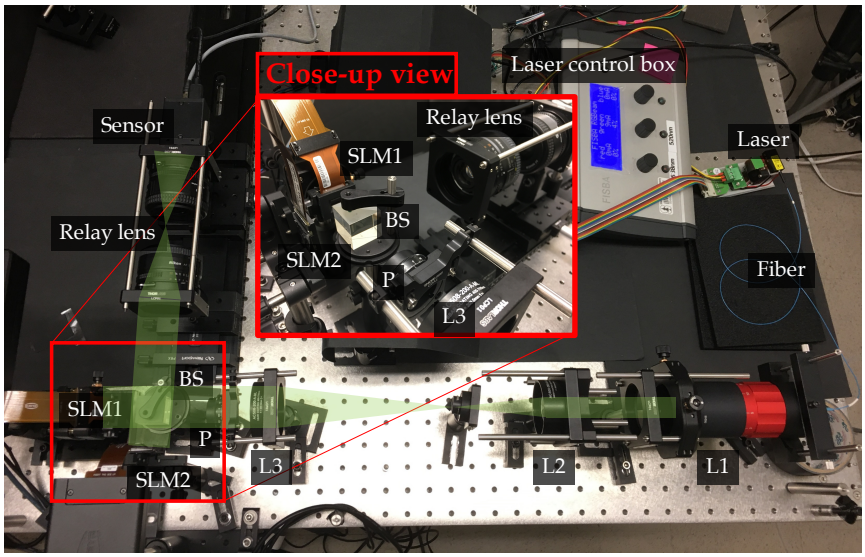

Fig. S1. Photograph of our prototype display. The light emitted from the single-mode fiber is collimated (L1, L2, L3) and polarized (P), before arriving at the beam splitter (BS) and SLMs (SLM1, SLM2). The modulated wave is imaged via 4f system (Relay lens) to the camera (Sensor).

B. Software Implementation

We implemented all CGH algorithms in PyTorch [1]. Pseudocode for SGD and camera-in-the-loop algorithms using two SLMs are outlined in Algorithms S1 and S2, respectively. The homographies used in these algorithms are found using procedures described in the Section C. The optimizations outlined in Algorithms S1 and S2 takes about 40 seconds and 500 seconds for 500 iterations, respectively. We refer to the supplementary material of [2] for pseudocodes for algorithms for the single SLM case. For all algorithms in this work, we set the learning rate to 0.01 for all phase variables, 0.001 for the scalar, and we use the L2 loss function. However, other loss functions could be used as well, for example a perceptual loss function [3], which may help bring out fine details as shown in Fig. S2.

B.1. Initialization

In all SGD-based optimizations for holographic displays, there may be infinitely many solutions that achieve a low residual or loss because it is non-convex. Thus, initialization plays a significant role on the performance as well as the resulting phase patterns. We use random initialization from

the range $(-0.5, 0.5)$ for the phase patterns, while the scalar s is initialized with the average value of the target amplitude. Note that we would want to pick a scalar that puts the reconstruction in the closest range to the target image. Thus, rather than optimizing it along with phase variables, one can just pick a reasonable initial scale, e.g. 0.75, and fix it during the optimization, or manually set it every iteration to minimize the residual loss, as $s_* = \arg\min_s \sum (s \cdot x_i - a_i)^2 = \frac{\sum x_i a_i}{\sum x_i^2}$, where x is the amplitude of the vectorized reconstruction and a is the target amplitude. These scalars would be effective resulting better quality, but should be calibrated per image/channel.

Algorithm S1. SGD Hologram Optimization with two SLMs

`ideal_p1/2`$(\cdot)$: propagation through idealized propagation models
`ideal_bp`$(\cdot)$: backpropagation through idealized models
$\mathcal{H}_h$: perspective transform with a homography h
`loss_bp`$(\cdot)$: backpropagation through loss function

foreach k **in** $1 \dots K$ **do**
  $\widehat{g} \leftarrow$ `ideal_p1`$(\phi_1) + \mathcal{H}_{H_1(H_2)^{-1}}($`ideal_p2`$(\phi_2))$ ▷ See Fig. S4 for H_1 and H_2.
  $\{\phi_1, \phi_2, s\} \leftarrow$ `ideal_bp`$($`loss_bp`$(\mathcal{L}(s \cdot |\widehat{g}|, a_{target})))$
return ϕ_1, ϕ_2

Algorithm S2. CITL Hologram Optimization with two SLMs

`ideal_p1/2`$(\cdot)$: propagation through idealized propagation models
`ideal_bp`$(\cdot)$: backpropagation through idealized models
$\mathcal{H}_h$: perspective transform with a homography h
`camera_p`$(\cdot)$: camera captured intensity (raw mode) + homography
`loss_bp`$(\cdot)$: backpropagation through loss function
`replace`(s, t) : replace values of s with t, retain gradients from s

foreach k **in** $1 \dots K$ **do**
  $\widehat{g} \leftarrow$ `ideal_p1`$(\phi_1) + \mathcal{H}_{H_1(H_2)^{-1}}($`ideal_p2`$(\phi_2))$ ▷ See Fig. S4 for H_1 and H_2.
  $|g| \leftarrow$ `replace`$(|\widehat{g}|, \sqrt{$`camera_p`$(\phi_1, \phi_2)})$
  $\{\phi_1, \phi_2, s\} \leftarrow$ `ideal_bp`$($`loss_bp`$(\mathcal{L}(s \cdot |g|, a_{target})))$
return ϕ_1, ϕ_2

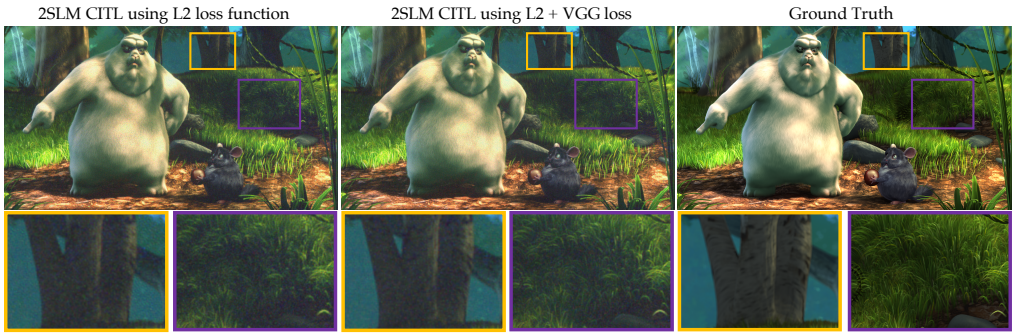

Fig. S2. Captured results using L2 loss and L2 loss with perceptual features. We set $\lambda_p = 0.025$ as the relative weight on the perceptual loss component, which is the output of the first five layers of the VGG16 network [4].

C. Homography Calibrations

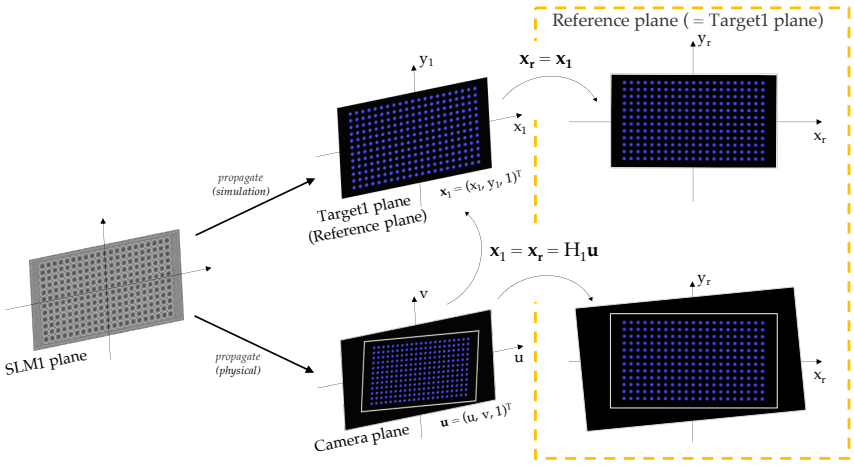

Fig. S3. Homography relationship for the single SLM setup

In this section, we provide background information of transformations between different coordinate frames that we use in this paper. Typically, there are two planar coordinate systems that come up in the CITL optimization.

First, we have a *target plane*, where the reconstructed image is formed by simulated propagation from the SLM plane. Note that the simulated images on this plane are always perfectly aligned with the ground truth image, assuming a proper propagation model. Thus we set this plane as a *reference plane* as well, where all camera images should be transformed to before being summed or being put into the loss function. The points of the target plane are denoted as $\mathbf{x_1} = (x_1, y_1, 1)$.

Next, we have a *camera plane*, on which a camera-captured image lies. Since this plane is formed by projecting the optically reconstructed image onto the camera sensor plane, this plane and the target plane form a projective geometry and there exists a transformation between two planes as a form of *perspective transformation*, or a *homography*, which is a general non-singular linear transformation of homogeneous coordinates:

$$\mathbf{x_1} = H_1\mathbf{u}, \tag{S1}$$

Here, $\mathbf{u}$ is a homogeneous coordinate of the camera image plane, and H_1 is a homography 3×3 matrix.

C.1. Definition of homogeneous coordinates

A point in Euclidean 2-space is represented by an ordered pair of real numbers, (x, y). We may add an extra coordinate to this pair, giving a triple $\mathbf{x} = (x, y, 1)$, that we declare to represent the same point. We can go back and forward from one representation of the point to the other, simply by adding or removing the last coordinate, so that (kx, ky, k) represents the same point as well, for any non-zero value k. These triples are called the homogeneous coordinates of the point [5].

In the following sections, We may index images with homogeneous coordinates: $I[\mathbf{u}] = I[u, v]$, for $\mathbf{u} = (u, v, 1)$. Namely, with two planes described in Eq. S1, the image at the other plane can be found as $I[H_1\mathbf{u}]$. Note that we usually process discrete images, while we may get some real-valued coordinates due to this transformation between image planes. For those real-valued indices, we use 2D bilinear interpolation.

C.2. Calibration procedure with single SLM setup

First, we consider single SLM case in [2] shown in Fig. S3. The CITL optimization (training) involves a loss function that takes a camera-captured image and a target image (or reconstruction) and we must ensure that these two images lies on the same coordinate frame.

We first display an SLM pattern that creates $21 \times 12 (= 252)$ circle grid patterns both at target and camera planes as in Fig. S3 to find an invertible projective transformation H_1, which relates the target plane $\mathbf{x_1}$ to the camera plane $\mathbf{u}$ as $\mathbf{x_1} = H_1\mathbf{u}$. Note that the reference coordinates of center of circle at the target plane are already known, because they are supposed to be fit in

a rectengular grid, for example, $(101, 161), (181, 161), \ldots, (980, 161), \ldots, (980, 1760)$. Then, H_1, which has 8 degrees of freedom can be found by minimizing the error such as $\sum_{i=1}^{252} (\mathbf{x_{1,i}} - H_1 \mathbf{u_i})^2$, where $\mathbf{x_{1,1}}, \ldots, \mathbf{x_{1,252}}$ are the reference coordinates on the target plane and $\mathbf{u_{c1}}, \ldots, \mathbf{u_{c252}}$ are detected coordinates of center of circles on the camera plane.

C.3. Homography relationships in dual SLM setup

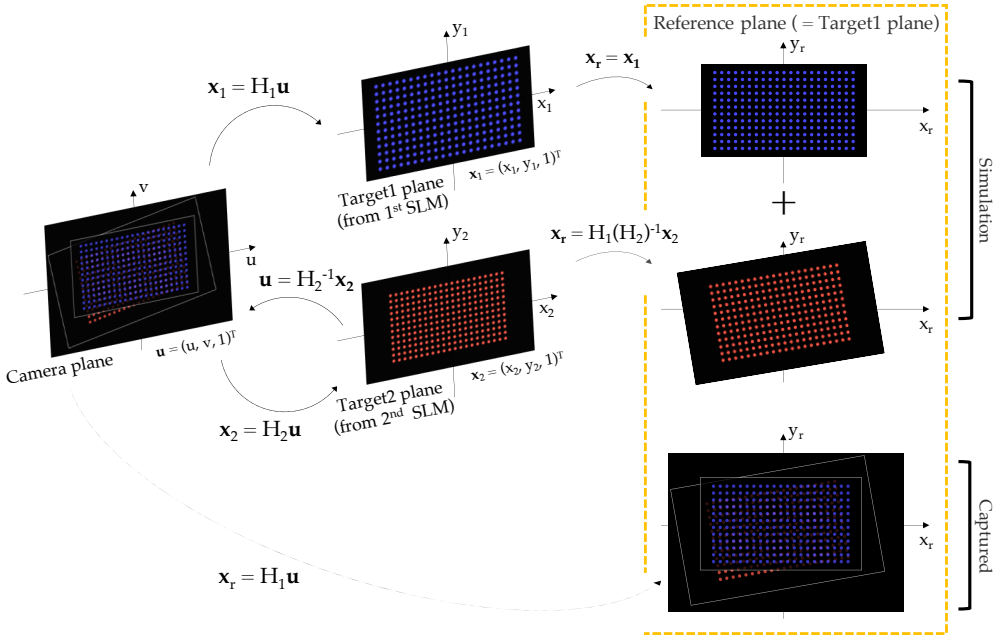

Fig. S4. Homography for the dual SLM setup. Note that $H_1 = H_2$ in the ideal case with the identical SLMs and the perfect align.

If we use additional SLM, there would be additional target plane (*Target2 plane*, $\mathbf{x_2}$) for the another propagation from the second SLM as shown in Fig. S4. Note that we can get a homography from each target plane to the camera plane, H_1 or H_2, as in Section C.2 by alternatingly displaying the dot pattern on one SLM and displaying a pattern that creates black target image on the other SLM.

We may align two SLMs after displaying dot patterns for each SLM at the same time and matching them in the camera plane. However, it is almost impossible to perfectly match them since it needs micron-scale accuracy. Thus, there must be subtle misalignment between them, as exaggerated in the camera plane in Fig. S4. To simulate the interference of wave fields from each SLM precisely, we need to consider the relative positioning of two SLMs before summing them up. As in Section C.2, we set the reference plane, where the summation of two fields or loss function is calculated, as the target plane of the first SLM (*Target1 plane)*. The warping from the *Target1 plane* or the camera plane to the reference plane is the same as described in Section C.2. Once we get two homographies relating target planes to the camera plane, we can get transformation from one target plane to the other target plane by using the camera plane as an intermediate plane. In other words, we can transform the coordinates on *Target2 plane* to the coordinates on *Target1 plane (reference plane)* as $\mathbf{x_1} = H_1 H_2^{-1} \mathbf{x_2}$.

2. ADDITIONAL RESULTS

This section includes additional captured results in support of those shown in the paper for different target images (Figs. S5 and S6). Michelson Holography shows the best results in all cases.

Fig. S5. Additional captured results with various methods. The included numbers are the PSNR value with respect to the target image.

Fig. S6. Additional captured results with various methods. The included numbers are the PSNR value with respect to the target image.

REFERENCES

1. A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in neural information processing systems*, (2019), pp. 8026–8037.
2. Y. Peng, S. Choi, N. Padmanaban, and G. Wetzstein, "Neural holography with camera-in-

the-loop training," ACM Trans. Graph. (SIGGRAPH Asia) (2020).
3. J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *ECCV,* (2016), pp. 694–711.
4. K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *CVPR,* (2014).
5. R. Hartley and A. Zisserman, *Multiple view geometry in computer vision* (Cambridge university press, 2003).