Neural 3D Holography: Learning Accurate Wave Propagation Models for 3D Holographic Virtual and Augmented Reality Displays—Supplemental Material

SUYEON CHOI*, MANU GOPAKUMAR*, and YIFAN PENG, Stanford University
JONGHYUN KIM, NVIDIA and Stanford University
GORDON WETZSTEIN, Stanford University

ACM Reference Format:

This supplementary document includes implementation details of both VR and AR display prototypes, complementary derivations related to wave propagation and optimization models of our neural 3D holography, and additional experimental results of both holographic display prototypes. Refer also to the supplementary video for better visualization.

Here we list the abbreviations and notations used across this document.

Authors’ addresses: Suyeon Choi*, Manu Gopakumar*, Yifan Peng, Stanford University; Jonghyun Kim, NVIDIA and Stanford University; Gordon Wetzstein, Stanford University.

*denotes equal contribution.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2021 Association for Computing Machinery.
0730-0301/2021/12-ART $15.00
https://doi.org/

S1 ADDITIONAL DETAILS ON HARDWARE SETUPs

In this section, we describe the hardware implementations of our two benchtop 3D holographic display prototypes. Figure S1 and Figure S2 show the system schematic and photographs of our VR and AR setups, each of them including a display and a capture unit, that are connected under a closed-loop framework. Specifically, the SLMs used in our VR and AR prototypes are Holosye Leto and Pluto phase-only LCoS devices, both with a resolution of 1,920 × 1,080 and a pixel pitch of 6.4 μm and 8.0 μm, respectively. This set of device provides a bit depth of 8 bits and a diffraction efficiency of over 80%.

For the VR setup, the laser is a FISBA RGBeam fiber-coupled module with three optically aligned laser diodes with a maximum output power of 50 mW. The measured wavelengths are 636.4, 517.7, and 440.8 nm. In our implementation, color images are captured as separate exposures for each wavelength and then cast in post-processing. For the AR setup, the laser is a Laser Quantum gem-532 free-space module at the wavelength of 532.0 nm. We additionally couple this laser output into a single-mode fiber so as to enable a reasonably good collimated beam shape later on.

Other components include the neutral density filter (Thorlabs NE510A), the linear polarizer (Thorlabs WP25M-VIS), the collimating lenses, the relay imaging lenses, the filtering iris, and the beam splitter (Thorlabs BS016) are shown in Figure S2. Refer to the caption for specifications of each optical component. In particular, for the AR setup, we use a micro-prisms-based lightguide (LLVision LEION) as the optical combiner. All images are captured with a FLIR Grasshopper3 2.3 MP color USB3 vision sensor through a Canon EF 50mm lens for the VR setup and a Canon EF 35mm lens for the AR setup. The SLM, Canon lens, and sensor are synchronized via Arduino (Uno SMD) controller to enable programmable varifocal display and acquisition. As such, the multi-plane holographic images are acquired to form the training dataset.
S2 PER-PIXEL LOOK UP TABLE CALIBRATION

In this section, we describe details of our approach to calibrate a spatially varying lookup table (SVLUT) for our SLM that is used for all methods shown in this paper.

Offset Sensitivity for Double-phase Method. Note that the double-phase method encodes a complex-valued wave field into the sum of two phase-only fields as:

\[ ae^{i\phi} = \frac{1}{2} e^{i(\phi + \cos^{-1}a)} + \frac{1}{2} e^{i(\phi - \cos^{-1}a)}, \]

where \( a \) is the normalized amplitude (0 < \( a \) < 1) and \( \phi \) is the phase value. Then, one would interlace two phase values after discarding every other pixel in the axis and form a checkerboard-like pattern as described by Maimone et al. [2017]. Eq. 1 also can be expressed by two phase values on the SLM \( \phi_1 = \phi + \cos^{-1}a \) and \( \phi_2 = \phi - \cos^{-1}a \) as:

\[ a = \cos \left( \frac{\phi_1 - \phi_2}{2} \right), \quad \phi = \frac{\phi_1 + \phi_2}{2}. \]

In Eq. 2, note that the amplitude of field is directly described by the offset around the mean phase value for adjacent pixels, and this makes DPAC more sensitive to the actual phase modulation by the SLM than other methods. Specifically, ensuring a \( \pi \) phase offset for adjacent pixels is the only way to make zero-amplitude. Even when the global lookup tables calibrated by the manufacturer are used, the slight inaccuracies in this phase offset results in low contrast as shown in several recent works [Chakravarthula et al. 2019, 2020; Peng et al. 2020].

Spatially Varying Lookup Table Calibration. As a result, we propose to calibrate a spatially varying lookup table for the SLM. Note that interferometric or diffraction-based approaches can be used and prior work has noted that spatially varying response of SLM can significantly affect the hologram quality [Engström et al. 2013]. In our approach, instead of constructing real lookup tables with 256 entries or fitting them in some polynomials [Dai et al. 2019; Engström et al. 2013], we train a 3 layer, 64 features multilayer perceptron (MLP) with two spatially varying latent codes per pixel and train it in a camera-in-the-loop manner after focusing the camera at SLM plane. Note that this MLP is used as a module in Neural Holography [Peng et al. 2020] but we use it in an inverse fashion: While the MLP of Neural Holography is used to model the physical non-linear response, our MLP is used to compensate the physical non-linear response so that we can modulate the desired phase on the SLM. Specifically, let \( f \) be the conversion on the SLM between the input of 8bit image \( V_i \in \{0, 1, 2, \cdots, 255\} \) and output of actual
modulated phase $\phi_{\text{real}} \in \mathbb{R}$:

$$\phi_{\text{real}} = f(V_i).$$

(3)

Note that this $f$ includes the calibrated lookup table operation commonly given by the manufacturer, as well as the physical voltage-to-phase conversion. In the ideal case, $f$ should be the linear function such as $f(x) \approx \frac{255}{\pi} x$ so that we can convert the desired phase $\phi_{\text{desired}} \in [0, 2\pi]$ into some 8bit imemap $V_i$ as $V_i = \frac{255}{\pi} \phi_{\text{desired}}$, which allows to modulate the desired phase delay:

$$\phi_{\text{actual}} = f\left(\frac{255}{\pi} \phi_{\text{desired}}\right) \approx \phi_{\text{desired}}.$$  

(4)

However, this is not the case and the phase response on the SLM $f$ is non-linear and even spatially variant. To consider this, we calibrate our own lookup table MLP that maps the desired phase $\phi_{\text{desired}}$ to the input value as $V_i = \text{MLP}(\phi_{\text{desired}})$ to approximately invert the function $f$ to achieve:

$$\phi_{\text{actual}} = f\left(V_i\right) = f\left(\text{MLP}(\phi_{\text{desired}})\right) = f\left(f^{-1}(\phi_{\text{desired}})\right) = \phi_{\text{desired}}.$$  

(5)

For an SLM with the resolution of $M \times N$, we display a checkerboard phase pattern $\phi_{\text{desired}} \in \mathbb{R}^{M \times N}$ which consists of only two values, $\phi_{\text{desired},1}$ and $\phi_{\text{desired},2}$, in our holographic prototype with the appropriate filter for DPAC. With our camera focused at the SLM, this results in a plain image corresponding to the spatially variant modulated phase $\phi_{\text{actual}} \in \mathbb{R}^{M \times N}$ where adjacent complex values on the SLM are blurred to form the amplitude as described in Equation 2. Similarly, we can also simulate the amplitude which would ideally be produced by just filtering out the high frequency of the SLM fields when the modulation is $\phi_{\text{desired}}$. Using this ideal amplitude as a target, we penalize the captured amplitude to be the target amplitude and backpropagate this loss through to the weights of our spatially varying lookup table (MLP). To do this, we use the same gradient replacing technique as in the camera-in-the-loop implementation [Peng et al. 2020]. To account for the spatially varying response, we add two latent codes per low resolution block size of $27 \times 48$ and upsample these codes by 40 to match the SLM resolution. These latent codes are then concatenated with the phase values to form the input of the MLP. We use an $\ell_1$ loss with a batch size of 4, a learning rate of $1e^{-4}$ and run 20,000 iterations for each channel. The resulting lookup tables are shown in Fig. S4. As shown in Fig. S3, the use of this spatially varying lookup table greatly increases the contrast for the double phase method, while also providing benefits to other methods such as SGD.

### S3 ADDITIONAL DETAILS ON SOFTWARE

In this section, we describe additional software implementation details including the procedures of different multi-plane phase generation algorithm.

**Homography Calibration.** As part of our calibration, we apply a planar homography to accurately register the captured images to the ground-truth images. Using the VR setup as the representative illustration, we compute our homography using a target binary pattern consisting of $22 \times 13$ white dots with a spacing of 80 pixels between the centers of neighboring dots. This produces a region of interest which has a resolution of $1,680 \times 960$ pixels. The baseline phase holograms for these target binary patterns are generated for all three color channels using a gradient descent CGH algorithm with a sufficient number of iterations. For each channel, the resulting hologram is displayed on the SLM, and its reconstruction image is captured by the sensor through the auxiliary optics. From this, a $3 \times 3$ homography matrix can be derived to account for the small warp between the captured plane and the simulated plane. Note, this operation can be plugged into the training pipeline before the start of every epoch to account for any experimental condition drift during a prolonged training process, but we observed our setup is stable enough with one-time homography calibration. For additional details, we refer the interested reader to Peng et al. [2020].

**Double-phase Method.** For the double-phase method (DPM), the phase at each of the target planes in a multi-plane scene is set to be $(2\pi \cdot z^{(j)})/\lambda$, where $z^{(j)}$ is the distance from the SLM to the sampled plane and $\lambda$ is the wavelength of illumination. This target phase initialization as described by Maimone et al. [2017] results in smooth object phase at the SLM plane which improves the performance of the double phase method. The multiple target planes with this phase initialization are propagated back to the SLM plane using ASM propagation. At the SLM, the wavefront is optionally blurred.

![Fig. S4. Calibrated spatially varying lookup tables for red, green, and blue. The solid line indicates mean value of all look up table at each pixel and blue shades show the variance.](image)
with a Gaussian kernel to implement the anti-aliased double phase method (AA-DPM), proposed by Shi et al. [2021], which reduces artifacts due to undersampling and phase wrapping. In the paper and throughout the supplement, we achieved best results by using AA-DPM with a preblur Gaussian kernel with $\sigma = 0.75$.

Finally, this complex valued wavefront is represented with phase-only modulation using double phase amplitude coding. With double phase amplitude coding, superpixels, composed of 2-by-2 adjacent SLM pixels, are used to represent complex values. In each 2-by-2 superpixel, one diagonal is assigned to the $\phi_1$ value and the other diagonal is assigned to the $\phi_2$ value defined in Equation 6 where $a$ and $\phi$ are the amplitude and phase of the desired complex valued wavefront. This phase modulation pattern blurs together to reproduce the desired SLM wavefront as shown in Equation 1.

$$
\begin{align*}
\phi_1 &= \phi - \cos^{-1} a \\
\phi_2 &= \phi + \cos^{-1} a
\end{align*}
$$

The range of phases produced by this procedure is centered around $\pi$ before being wrapped to fit in a $0$ to $2\pi$ range. The wrapping is necessary because our SLM can only produce a range of $2\pi$ phase modulation. The centering reduces the amount of wrapping to minimize the resulting artifacts.

As discussed in the prior section, the performance of the double phase method is very sensitive to the calibration of the SLM and its performance degrades for content which is further from the SLM. The optimization method, we seek to minimize the masked multiplane loss in Equation 7 to reproduce the target amplitude at each of the target planes.

$$
\begin{align*}
\minimize_{\{\phi, s, a_{\text{target}}, m\}} & L_{\text{SGD}}(\phi, s, a_{\text{target}}, m), \\
\text{where } L_{\text{SGD}} \text{ is defined as } & \\
L_{\text{SGD}}(\phi, s, a_{\text{target}}, m) &= \sum_{j=1}^{J} \left| \left(s \cdot f(e^{i\phi}, z^{(j)}) | - a_{\text{target}}^{(j)} \right) \circ m^{(j)} \right|^2.
\end{align*}
$$

This loss is minimized with $f = f_{\text{ASM}}$ for SGD-ASM and $f = f_{\text{CNNpropCNN}}$ for SGD-CNNpropCNN. For $f_{\text{ASM}}$, frequencies larger than half of maximum frequency are filtered out in the frequency domain to account for optical filtering in the setup. The optimization is performed using an Adam optimizer with $\phi$ initialized to uniform random phase from a range of $[-1.25, 1.25]$. For the results throughout the paper, the Adam optimizer learning rate is set to $5e^{-3}$ and run for 1000 iterations. The full algorithm is summarized in Algorithm S1.

**Alternating Direction Method of Multipliers.** The alternating direction method of multipliers method (ADMM) is used to minimize the masked multiplane loss with prior in Equation 8 to reproduce the scene content at each of the target planes and to enforce target phase smoothness. The use of the ADMM optimizer is motivated by the difficulty the Adam optimizer has with enforcing the Laplacian phase sparsity. The difference in performance between the ADMM and Adam optimizer are illustrated qualitatively in Figure S7 and quantitatively in Figure S8.

$$
\begin{align*}
\minimize_{\{\phi, s\}} & \sum_{j=1}^{J} \left| \left(s \cdot f_{\text{CNNpropCNN}} \left(e^{i\phi}, z^{(j)} \right) | - a_{\text{target}}^{(j)} \right) \circ m^{(j)} \right|^2 \\
& + \gamma \sum_{j=1}^{J} \left| \Delta \phi \left(f_{\text{ASM}} \left(\text{CNN}_{\text{SLM}} \left(e^{i\phi}, z^{(j)} \right) \right) \circ m^{(j)} \right|_1, \tag{8}
\end{align*}
$$

![Fig. S5. Comparison of results generated with DPAC at different distances.](image)

![Fig. S6. Quantitative comparison of DPM and AA-DPM at different distances with different sized Gaussian kernels used for AA-DPM. Preblur with a $\sigma = 0$ kernel references the conventional double phase method.](image)
To calculate $\Delta \Phi$ we use the signed angular distance between wrapped phase samples. By introducing a slack variable, $\xi_j$, this can be written in the standard ADMM form as in Equation 9:

$$\begin{align*}
\text{minimize} \quad & \sum_{j=1}^{J} \left\| s \cdot [f_{\text{CNNpropCNN}} \left( e^{i\phi}, z^{(j)} \right)] - a^{(j)}_{\text{target}} \right\|_2^2 \\
& + \gamma \sum_{j=1}^{J} \| \xi_j \|_1, \\
\text{subject to} \quad & \Delta \Phi \left( f_{\text{ASM}} \left( \text{CNN}_{\text{ASM}} \left( e^{i\phi} \right), z^{(j)} \right) \right) \circ m^{(j)} - \xi - \zeta = 0. \quad (9)
\end{align*}$$

With augmented Lagrangian ADMM, this produces the update steps in Equation 10 for each ADMM iteration.

1. $\phi, s \leftarrow \arg\min_{(\phi, s)} \mathcal{L}_{\text{inner}}(\phi, s, u, \zeta, a_{\text{target}}, m), \quad (10)$

where

$$\mathcal{L}_{\text{inner}}(\phi, s, u, \zeta, a_{\text{target}}, m) = \sum_{j=1}^{J} \left\| s \cdot [f_{\text{CNNpropCNN}} \left( e^{i\phi}, z^{(j)} \right)] - a^{(j)}_{\text{target}} \right\|_2^2 \\
+ \frac{\rho}{2} \left\| \Delta \Phi \left( f_{\text{ASM}} \left( \text{CNN}_{\text{ASM}} \left( e^{i\phi} \right), z^{(j)} \right) \right) \circ m^{(j)} - \zeta - \xi + u_j \right\|_2^2.$$

2. $\xi_j \leftarrow \arg\min_{\xi_j} \left\{ \frac{\rho}{2} \left\| \Delta \Phi \left( f_{\text{ASM}} \left( \text{CNN}_{\text{ASM}} \left( e^{i\phi} \right), z^{(j)} \right) \right) \circ m^{(j)} - \xi_j + u_j \right\|_2^2 + \gamma \| \xi_j \|_1,$

3. $u_j \leftarrow u_j + \Delta \Phi \left( f_{\text{ASM}} \left( \text{CNN}_{\text{ASM}} \left( e^{i\phi} \right), z^{(j)} \right) \right) \circ m^{(j)} - \xi_j.$

The $\phi$ update of Equation 10 is minimized with an inner loop of gradient descent performed with an Adam optimizer. The $\xi$ update is minimized with soft thresholding as in Equation 11. The full algorithm is summarized in Algorithm S3.

$$\xi_j \leftarrow S_{\delta / \rho} \left( \Delta \Phi \left( f_{\text{ASM}} \left( \text{CNN}_{\text{ASM}} \left( e^{i\phi} \right), z^{(j)} \right) \right) \circ m^{(j)} + u_j \right). \quad (11)$$

For the results in the paper, the $\phi$ is initialized to uniform random phase from a range of $[-1.25, 1.25]$. The Adam optimizer for the $\phi$ update is run with a learning rate of 0.01 for 50 iterations. The ADMM optimizer is run for 50 iterations with $\rho = 0.5$ and $\gamma = 0.01$. With these parameters, we are able to successfully produce our desired target amplitude while enforcing piecewise smoothness of the phase as illustrated by the comparison between SGD-CNNpropCNN and ADMM-CNNpropCNN in Fig. S9.

### Forward Model Training and Parameter Counts

We trained all models with the same procedure outlined in Algorithm S2. The initial phase pool is made of 1,000 SGD-optimized phases and 100 DPAC-generated phases for each plane, leading to 8,800 phases in total for each channel. Two different phase pools of 1,100 phases are also similarly generated for validation and test. The model with the best validation loss is applied for testing, and one additional phase pool generation for each model is performed. Most of them were converged within 30 epochs, which takes about 2 days. We use the $\ell_1$ loss, a learning rate of $5 \times 10^{-4}$, and a batch size of 1.

Our proposed forward models have a similar number of learnable parameters to the prior hardware-in-the-loop (HIL) model proposed by Chakravarthula et al. [2020]. The HIL model, propCNN, and CNNprop have 65 million parameters each, and CNNpropCNN has 67 million parameters. The prior neural holography model proposed by Peng et al. [2020], which was designed to learn a fewer handcrafted parameters, only has 8 million learned parameters, but it lacks the flexibility to fully model the captured behavior.

### S4 PSEUDOCODE FOR ALL ALGORITHMS

We provide the pseudocode for all algorithms, including the SGD phase retrieval (Algorithm S1), the forward model training (Algorithm S2), and the ADMM solver for phase regularization (Algorithm S3).
Algorithm S1: SGD

\( K \): Number of SGD iterations
\( \alpha \): Learning rate
\( L_{\text{SGD}}(\phi, s, a_{\text{target}}, m) \): Loss for SGD iterations as defined in Equation 7

\( \phi \) starts from uniform random initialization.
\( m \) and \( a_{\text{target}} \) are set based on a multiplane decomposition of a target rgb\!d scene.

```
foreach k in 1 \ldots K do 
// Adam optimizer descent steps 
    \[ \phi \leftarrow \phi - \alpha \nabla_{\phi} L_{\text{SGD}}(\phi, s, u, \xi, a_{\text{target}}, m) \]
end 
return \( \phi \)
```

Algorithm S2: Forward model training

\( F \): Number of focal slices
\( \hat{g}_\theta(\cdot; d) \):
Our propagation model to be trained with propagation distance(s) \( d \)

\( |g|^2(\cdot; \{d_1, \ldots, d_F\}) \):
Captured focal stack in raw intensity with the camera
\( M \): Size of minibatch
\( N \): Number of training images
\( E \): Number of epochs
\( P \): Our dataset (phase)
\( C \): Our dataset (captured amplitudes; \( |\hat{g}(P)| \))
\( L_{\text{model}} \): L1 loss applied between predicted focal stack and captured focal stack defined as below:
\( L_{\text{model}} = \| |\hat{g}_\theta(\phi; \{d_1, \ldots, d_F\})| - |g(\phi; \{d_1, \ldots, d_F\})| \|_1 \)
\( L_{\text{phase}} \): Singleplane variant of SGD loss defined as below:
\( L_{\text{phase}} = \| s \cdot \hat{g}(\phi; d) - a_{\text{target}} \|^2 \)

\( \hat{g}_\theta \) starts from \( \hat{g}_{\text{ASM}} \) (via residual connection)

```
foreach e in 1 \ldots E do 
    // 1) Creating dataset. 
    if e == 1 or \( \hat{g}_\theta \) is converged then 
        \( P = \{\} \)
    foreach n in 1 \ldots N do 
        foreach d in \( d_1 \ldots d_F \) do 
            Initialize \( \phi_{n,d} \) with random phase
            foreach k in 1 \ldots K do 
                // Adam optimizer descent steps 
                \[ \phi_{n,d,k+1} \leftarrow \phi_{n,d,k} - \alpha \nabla_{\phi} L_{\text{phase}}(\hat{g}(\phi; d), a_{\text{target}}) \]
            For random \( k, d \) and \( n \), append \( \phi_{n,d,k} \) to \( P \)
        end 
    end 
    // 2) You can precapture \( C \) with \( P \) here (focal stack).
```

```
// 3) Training our network params with focal stack supervision.
foreach \( \phi, |g(\phi; \{d_1, \ldots, d_F\})| \) in zip(\( P, C \)) do 
    // Adam optimizer descent steps 
    \[ \theta \leftarrow \theta - \beta \nabla_{\theta} L_{\text{model}}(\hat{g}_\theta(\phi; \{d_1, \ldots, d_F\}), |g(\phi; \{d_1, \ldots, d_F\})|) \]
end 
return \( \hat{g}_\theta \)
```
Algorithm S3: ADMM

\( K \) : Number of ADMM iterations  
\( K_{inner} \) : Number of SGD iterations for \( \phi \) update  
\( \alpha \) : Learning rate of SGD iterations for \( \phi \) update  
\( \mathcal{L}_{inner}(\phi, s, u, \zeta, a_{target}, m) \) : Loss for SGD iterations as defined in \( \phi \) update of Equation 10

\( \phi \) starts from uniform random initialization.  
\( u \) and \( \zeta \) all start as zeros.  
\( m \) and \( a_{target} \) are set based on a multiplane decomposition of a target rgbd scene.

foreach \( k \) in 1 \ldots \( K \) do  
  foreach \( k_{inner} \) in 1 \ldots \( K_{inner} \) do  
    // Adam optimizer descent steps  
    \( \phi \leftarrow \phi - \alpha \nabla \phi \mathcal{L}_{inner}(\phi, s, u, \zeta, a_{target}, m) \)  
  end  
  foreach \( j \) in 1 \ldots \( J \) do  
    \( \zeta_j \leftarrow S_{\gamma \rho} \left( \Delta \Phi \left( f_{ASM} \left( \text{CNN}_{ASM} \left( e^{i\phi}, z^{(j)} \right) \right) \circ m^{(j)} + u_j \right) \right) \)  
  end  
  foreach \( j \) in 1 \ldots \( J \) do  
    \( u_j \leftarrow u_j + \Delta \Phi \left( f_{ASM} \left( \text{CNN}_{ASM} \left( e^{i\phi}, z^{(j)} \right) \right) \circ m^{(j)} - \zeta_j \right) \)  
  end

return \( \phi \)
S5 ADDITIONAL RESULTS OF THE VR PROTOTYPE
In this section, we present additional experimental results of the VR holographic display prototype in support of those shown in the paper.

S5.1 Additional 2D Results
For all 2D holographic image results shown in the paper and supplement, we positioned the target plane on which the image is shown and optically recorded at a distance of 4.4 mm from the SLM, which corresponds to a distance of 1.26 D or 0.80 m from the camera or user.

Figures S10–S12 show the 15 test images used for the quantitative evaluation shown in Table 1 in the main text and Supplemental Tables 1, 2. For each of these 15 images, we capture and compare experimental results using the following methods: our implementation of the double phase–amplitude coding method [Maimone et al. 2017] (see Sec. S3 for implementation details) (DPAC), an SGD-based phase retrieval algorithm using ASM wave propagation applied to the target amplitude [Peng et al. 2020] (SGD-ASM), the model-based approach proposed by Chakravarthula et al. [2020] (SGD-HIL model), the model-based approach proposed by Peng et al. [2020] (SGD-NH model), the camera-in-the-loop approach proposed by Peng et al. [2020] (CTTL-ASM), and three different variants of the proposed model-based approach using slightly different variants of our model, i.e., propCNN, CNNprop, and CNNpropCNN.

All of these approaches are directly comparable as they are all captured under the same experimental conditions. Note that our implementation of DPAC followed the description of two recent papers [Maimone et al. 2017; Shi et al. 2021] as closely as possible but despite lots of efforts on our end, we were not able to achieve the same quality of results these authors were able to demonstrate. This may be due to slight differences in the hardware setup, which (if mitigated) may also improve the results of all other methods. Therefore, we argue that this is a fair comparison under exactly comparable conditions.

Tables 1, 2 include quantitative evaluations using experimentally captured results with peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) metrics, respectively. We show PSNR/SSIM values for all 15 test images individually and also the average values. In all cases, the proposed model variants outperform all previous methods by a large margin. On average, our CNNpropCNN model beats the next best approach, CTTL-ASM, by almost 2 dB PSNR on average.

S5.2 Additional 3D Results
For all 3D holographic image results shown in the paper and supplement, we have positioned the target planes, on which the multipeptide images are shown and optically recorded, in a range of 0–2 D from the camera. Within this range, we use 8 planes in total, spaced at equal distances in dioptric space (i.e., user-perceived distances). We measured these distances to correspond to 0.0, 1.1, 2.1, 3.2, 4.4, 5.7, 7.0, and 8.2 mm away from the SLM physically. The inter-plane distance of 0.31 D thus corresponds to approx. the depth of field of the human eye [Campbell 1957; Marcos et al. 1999] and can therefore be considered approx. continuous in depth, as perceived by a human observer.

Figures S13–S21 show experimentally captured results of 4 multipeptide 3D scenes, each focused at 3 different distance. We compare experimental results with our implementation of the double phase–amplitude coding method [Maimone et al. 2017] (see Sec. S3 for implementation details) (DPAC), a conventional multipeptide SGD optimization that uses ASM wave propagation (SGD-ASM), the proposed multipeptide 3D model used together with a multipeptide SGD solver (SGD-proposed model), and the same proposed multipeptide 3D model used together with an ADMM solver (ADMM-proposed model) that enforces piecewise smooth phase constraints of the in-focus multipeptide images, as described in the paper. In these results, we see that our implementation of DPAC shows overall reasonably good quality for in-focus (red boxes) and out-of-focus (white boxes) parts of the scenes, although (despite our best efforts) the contrast is somewhat low. The SGD-ASM solver significantly improves the contrast over DPAC, but it is much more noisy in both in-focus and out-of-focus image regions. The proposed model adequately models the wave propagation from the SLM to all planes and a multipeptide SGD solver that constrains the in-focus parts of the target image (third column) achieves a good image quality with significantly reduced speckle and better image quality in these in-focus parts. However, because the out-of-focus behavior is unconstrained, as the wave field propagates away from the constrained in-focus parts, its unconstrained out-of-focus effects exhibit significant speckle.

Using the proposed piecewise smooth in-focus phase constraints mitigates this out-of-focus speckle behavior and results in the best in-focus and out-of-focus image quality.

Figure S22 shows additional ablations for our 3D results and model. Specifically, these figures show a composite image that combines only the in-focus parts of all recorded images as well as captured images at 3 different focus settings for one scene in the respective rows. In the columns, we compare SGD-ASM with different variants of our 3D model, including propCNN, CNNprop, and our final model CNNpropCNN. Both qualitatively and quantitatively (PSNR/SSIM in boxes), our final model outperforms SGD-ASM and other variants of our model. Figure S12 shows only the composite images with the in-focus parts of several different scenes. In this qualitative and quantitative comparison, one would expect our 3D model used with either SGD or ADMM using the additional phase constraints to perform roughly equivalent, because these results only show the in-focus parts of the scene and both approaches use the same wave propagation model. Yet, small differences in the solvers and manually tuned hyperparameters lead to minor discrepancies between the PSNR/SSIM metrics. Qualitatively, these two methods perform roughly equal with both showing significant improvements over alternative methods.

Table 3 quantitatively evaluates and ablates our model in various scenarios for only the green laser. The upper block shows a 2D variant of our CNNpropCNN model trained for a single plane at 4.4 mm from the SLM. We compare ASM, the HIL model, the NH
model, and the three variants of our model. On the training set, our model outperforms both HIL and NH by more than 8 dB PSNR, which is a significant improvement. To demonstrate that this is not due to overfitting, we evaluate the 2D model performance using the DIV2K test set of 1,100 images. The improvement of our model over these previous models is still about 7 dB for the test set of unseen images, confirming that our model is indeed better and generalizes from training to test set. The next block of the table shows different variants of our 3D multiplane model trained on 7 of the 8 planes. We intentionally left out the 3rd plane from the SLM at 0.6 D or 2.1 mm from the training procedure. Again, our CNNpropCNN model shows the best quality for both training set and test sets. Interestingly, the quality of our 2D and 3D models evaluated at the same plane for both test and training sets does not drop significantly, about 0.9 dB for the test set, indicating that our model naturally extends from 2D to 3D. We also evaluate the performance of this multiplane model on the held-out plane, which was not part of the training, to understand the performance of our model and its capability to generalize to planes in-between the training planes. All of our 3D model variants that use a CNN on the SLM, i.e., CNNprop and CNNpropCNN, generalize much better to the held out plane than the variant of our 3D model that only uses a CNN at the target planes. The same trends are also observed when we use the same model that was trained on 7 out of the 8 planes and evaluate it, for both training and test set, on all of these 7 planes (lower block of Table 3).

Figures S24 – S32 qualitatively evaluate and ablate our model for scenes in the test set for all colors. In Fig. S24, we show qualitative reconstruction and errors for the models trained on a single intensity plane. At odd rows are reconstruction results from unseen phases. Even rows visualize absolute errors averaged over all wavelengths. The numbers on the right indicate the PSNR numbers in dB. Errors are visualized in log scale and the colorbar is on the right. This figure demonstrates that shows the best quality for scenes in the test dataset. In Figs. S25 – S32, we show qualitative reconstruction and errors of different models on captured test dataset. Note that the fifth row (red box) is the plane supervised in single-plane trained cases (From the first column to the fifth column). The sky box clearly shows that a CNN at the target plane does not generalize to other planes if only trained on a single plane. The third row is the held-out plane, which is not trained on even in multi-plane trained cases.

Overall, these experiments demonstrate that the 2D variant of our model is better than previously proposed wave propagation models, which are limited to 2D settings, and ours is also the only one that generalizes to 3D multiplane settings. Moreover, we tested several variants of our model and note that a CNN correcting the wave field on the SLM is necessary for good generalization behavior in between planes of the training set. Yet, an additional CNN on the target planes can help correct for plane-specific artifacts, and the combination of CNNs on both SLM and target planes is the best.

S5.3 Robustness to Possible Viewpoint Shifts
In Figure S23, we intuitively validate the robustness of our image synthesis to possible viewpoint shifts on our VR holographic display prototype. The camera is manually translated in horizontal from right to left, and then back, for a few millimeters. We observe no noticeable degradation in image quality over the viewpoint shifts.
Fig. S10. Additional experimentally captured 2D results. From left: DPAC, SGD-ASM, SGD-HIL model, SGD-NH model, CITL-ASM, SGD-propCNN, SGD-CNNprop, SGD-CNNpropCNN. Source images by Ana Blazic Pavlovic/Shutterstock, Blender Foundation, and [Agustsson and Timofte 2017]
Fig. S11. Additional experimentally captured 2D results. From left: DPAC, SGD-ASM, SGD-HIL model, SGD-NH model, CITL-ASM, SGD-propCNN, SGD-CNNprop, SGD-CNNpropCNN. Source images by Blender Foundation, [Agustsson and Timofte 2017; Kim et al. 2013]
Fig. S12. Additional experimentally captured 2D results. From left: DPAC, SGD-ASM, SGD-HIL model, SGD-NH model, CITL-ASM, SGD-propCNN, SGD-CNNprop, SGD-CNNpropCNN. Source images by Blender Foundation, [Agustsson and Timofte 2017; Kim et al. 2013]
Table 1. PSNR metrics of all captured 2D results. Among all the methods, the proposed model, CNNpropCNN, achieves the highest PSNR.

<table>
<thead>
<tr>
<th>Methods (algorithm-propagation operator)</th>
<th>DPAC</th>
<th>SGD-ASM</th>
<th>SGD-NH</th>
<th>SGD-HIL</th>
<th>crtn-ASM</th>
<th>propCNN</th>
<th>CNNprop</th>
<th>CNNpropCNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>15.9</td>
<td>16.8</td>
<td>19.0</td>
<td>19.0</td>
<td>19.5</td>
<td>20.3</td>
<td>21.1</td>
<td><strong>21.4</strong></td>
</tr>
<tr>
<td>#2</td>
<td>15.8</td>
<td>18.1</td>
<td>19.1</td>
<td>18.9</td>
<td>20.3</td>
<td>20.5</td>
<td>22.1</td>
<td><strong>22.3</strong></td>
</tr>
<tr>
<td>#3</td>
<td>17.1</td>
<td>18.6</td>
<td>19.8</td>
<td>19.6</td>
<td>21.0</td>
<td>20.9</td>
<td>22.7</td>
<td><strong>22.8</strong></td>
</tr>
<tr>
<td>#4</td>
<td>17.7</td>
<td>18.8</td>
<td>21.1</td>
<td>20.7</td>
<td>22.4</td>
<td>22.3</td>
<td>23.9</td>
<td><strong>24.1</strong></td>
</tr>
<tr>
<td>#5</td>
<td>18.2</td>
<td>19.4</td>
<td>21.7</td>
<td>21.7</td>
<td>23.3</td>
<td>23.3</td>
<td>25.1</td>
<td><strong>25.5</strong></td>
</tr>
<tr>
<td>#6</td>
<td>15.1</td>
<td>16.3</td>
<td>16.3</td>
<td>16.9</td>
<td>17.6</td>
<td>18.0</td>
<td>19.1</td>
<td><strong>19.1</strong></td>
</tr>
<tr>
<td>#7</td>
<td>18.2</td>
<td>18.6</td>
<td>20.8</td>
<td>21.1</td>
<td>22.4</td>
<td>22.4</td>
<td>23.8</td>
<td><strong>23.8</strong></td>
</tr>
<tr>
<td>#8</td>
<td>19.4</td>
<td>19.8</td>
<td>21.1</td>
<td>21.3</td>
<td>21.9</td>
<td>22.0</td>
<td>23.0</td>
<td><strong>23.2</strong></td>
</tr>
<tr>
<td>#9</td>
<td>18.9</td>
<td>20.2</td>
<td>22.9</td>
<td>22.9</td>
<td>25.7</td>
<td>24.7</td>
<td>27.8</td>
<td><strong>28.2</strong></td>
</tr>
<tr>
<td>#10</td>
<td>16.3</td>
<td>17.5</td>
<td>19.4</td>
<td>19.2</td>
<td>19.9</td>
<td>20.6</td>
<td>21.8</td>
<td><strong>21.9</strong></td>
</tr>
<tr>
<td>#11</td>
<td>16.3</td>
<td>17.3</td>
<td>19.0</td>
<td>18.9</td>
<td>19.6</td>
<td>20.0</td>
<td>21.0</td>
<td><strong>21.1</strong></td>
</tr>
<tr>
<td>#12</td>
<td>17.0</td>
<td>18.6</td>
<td>19.9</td>
<td>19.8</td>
<td>20.8</td>
<td>21.1</td>
<td>22.5</td>
<td><strong>22.8</strong></td>
</tr>
<tr>
<td>#13</td>
<td>17.6</td>
<td>18.8</td>
<td>20.0</td>
<td>19.9</td>
<td>21.1</td>
<td>21.2</td>
<td>22.4</td>
<td><strong>22.6</strong></td>
</tr>
<tr>
<td>#14</td>
<td>17.7</td>
<td>18.3</td>
<td>20.1</td>
<td>20.0</td>
<td>21.0</td>
<td>21.1</td>
<td>22.5</td>
<td><strong>22.5</strong></td>
</tr>
<tr>
<td>#15</td>
<td>14.6</td>
<td>16.0</td>
<td>17.2</td>
<td>17.1</td>
<td>17.5</td>
<td>18.2</td>
<td>19.2</td>
<td><strong>19.3</strong></td>
</tr>
</tbody>
</table>

**Avg.** | 17.0 | 18.2 | 19.8 | 19.8 | 20.9 | 21.1 | **22.7** |

Table 2. SSIM metrics of all captured 2D results. Among all the methods, ours achieves the highest SSIM.

<table>
<thead>
<tr>
<th>Methods (algorithm-propagation operator)</th>
<th>DPAC</th>
<th>SGD-ASM</th>
<th>SGD-NH</th>
<th>SGD-HIL</th>
<th>crtn-ASM</th>
<th>propCNN</th>
<th>CNNprop</th>
<th>CNNpropCNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>0.51</td>
<td>0.52</td>
<td>0.64</td>
<td>0.65</td>
<td>0.67</td>
<td>0.69</td>
<td>0.73</td>
<td><strong>0.74</strong></td>
</tr>
<tr>
<td>#2</td>
<td>0.65</td>
<td>0.61</td>
<td>0.58</td>
<td>0.57</td>
<td>0.66</td>
<td>0.67</td>
<td><strong>0.74</strong></td>
<td>0.74</td>
</tr>
<tr>
<td>#3</td>
<td>0.67</td>
<td>0.64</td>
<td>0.68</td>
<td>0.67</td>
<td>0.78</td>
<td>0.71</td>
<td>0.79</td>
<td><strong>0.82</strong></td>
</tr>
<tr>
<td>#4</td>
<td>0.69</td>
<td>0.58</td>
<td>0.63</td>
<td>0.62</td>
<td>0.75</td>
<td>0.69</td>
<td>0.78</td>
<td><strong>0.79</strong></td>
</tr>
<tr>
<td>#5</td>
<td>0.74</td>
<td>0.65</td>
<td>0.73</td>
<td>0.72</td>
<td>0.82</td>
<td>0.79</td>
<td>0.86</td>
<td><strong>0.87</strong></td>
</tr>
<tr>
<td>#6</td>
<td>0.62</td>
<td>0.53</td>
<td>0.48</td>
<td>0.54</td>
<td>0.59</td>
<td>0.59</td>
<td><strong>0.65</strong></td>
<td>0.65</td>
</tr>
<tr>
<td>#7</td>
<td>0.81</td>
<td>0.78</td>
<td>0.82</td>
<td>0.84</td>
<td>0.88</td>
<td>0.87</td>
<td>0.89</td>
<td><strong>0.90</strong></td>
</tr>
<tr>
<td>#8</td>
<td>0.85</td>
<td>0.81</td>
<td>0.84</td>
<td>0.85</td>
<td>0.88</td>
<td>0.86</td>
<td>0.88</td>
<td><strong>0.89</strong></td>
</tr>
<tr>
<td>#9</td>
<td>0.84</td>
<td>0.76</td>
<td>0.79</td>
<td>0.79</td>
<td>0.90</td>
<td>0.85</td>
<td>0.91</td>
<td><strong>0.92</strong></td>
</tr>
<tr>
<td>#10</td>
<td>0.64</td>
<td>0.67</td>
<td>0.73</td>
<td>0.73</td>
<td>0.78</td>
<td>0.78</td>
<td><strong>0.83</strong></td>
<td>0.83</td>
</tr>
<tr>
<td>#11</td>
<td>0.60</td>
<td>0.53</td>
<td>0.59</td>
<td>0.59</td>
<td>0.66</td>
<td>0.63</td>
<td><strong>0.69</strong></td>
<td>0.69</td>
</tr>
<tr>
<td>#12</td>
<td>0.58</td>
<td>0.51</td>
<td>0.53</td>
<td>0.54</td>
<td>0.64</td>
<td>0.63</td>
<td>0.72</td>
<td><strong>0.74</strong></td>
</tr>
<tr>
<td>#13</td>
<td>0.67</td>
<td>0.54</td>
<td>0.58</td>
<td>0.60</td>
<td>0.75</td>
<td>0.66</td>
<td>0.74</td>
<td><strong>0.77</strong></td>
</tr>
<tr>
<td>#14</td>
<td>0.72</td>
<td>0.64</td>
<td>0.72</td>
<td>0.72</td>
<td>0.80</td>
<td>0.77</td>
<td><strong>0.83</strong></td>
<td>0.83</td>
</tr>
<tr>
<td>#15</td>
<td>0.49</td>
<td>0.44</td>
<td>0.50</td>
<td>0.50</td>
<td>0.58</td>
<td>0.57</td>
<td>0.65</td>
<td><strong>0.66</strong></td>
</tr>
</tbody>
</table>

**Avg.** | 0.67 | 0.61 | 0.66 | 0.66 | 0.74 | 0.72 | 0.78 | **0.79** |
Fig. S13. Additional experimentally captured 3D results. From left: DPAC, SGD-ASM, the proposed model (CNNpropCNN) used with SGD, and the proposed model used with the ADMM solver enforcing additional phase constraints on the target planes. Source images by [Roberts and Paczan 2020]
Fig. S14. Additional experimentally captured 3D results. From left: DPAC, SGD-ASM, the proposed model (CNNpropCNN) used with SGD, and the proposed model used with the ADMM solver enforcing additional phase constraints on the target planes. Source images by [Roberts and Paczan 2020].
**Fig. S15.** Additional experimentally captured 3D results. From left: DPAC, SGD-ASM, the proposed model (CNNpropCNN) used with SGD, and the proposed model used with the ADMM solver enforcing additional phase constraints on the target planes. Source images by [Roberts and Paczan 2020]
Fig. S16. Additional experimentally captured 3D results. From left: DPAC, SGD-ASM, the proposed model (CNNpropCNN) used with SGD, and the proposed model used with the ADMM solver enforcing additional phase constraints on the target planes. Source images by [Roberts and Paczan 2020].
Fig. S17. Additional experimentally captured 3D results. From left: DPAC, SGD-ASM, the proposed model (CNNpropCNN) used with SGD, and the proposed model used with the ADMM solver enforcing additional phase constraints on the target planes. Source images by [Kim et al. 2013].
Fig. S18. Additional experimentally captured 3D results. From left: DPAC, SGD-ASM, the proposed model (CNNpropCNN) used with SGD, and the proposed model used with the ADMM solver enforcing additional phase constraints on the target planes. Source images by [Xiao et al. 2018]
Fig. S19. Additional experimentally captured 3D results. From left: DPAC, SGD-ASM, the proposed model (CNNpropCNN) used with SGD, and the proposed model used with the ADMM solver enforcing additional phase constraints on the target planes. Source images by [Roberts and Paczan 2020]
Fig. S20. Additional experimentally captured 3D results. From left: DPAC, SGD-ASM, the proposed model (CNNpropCNN) used with SGD, and the proposed model used with the ADMM solver enforcing additional phase constraints on the target planes. Source images by [Roberts and Paczan 2020].
Fig. S21. Additional experimentally captured 3D results. From left: DPAC, SGD-ASM, the proposed model (CNNpropCNN) used with SGD, and the proposed model used with the ADMM solver enforcing additional phase constraints on the target planes. Source images by Blender Foundation.
Fig. S22. Ablation study with captured results of a 3D scene. From left: SGD with ASM, SGD with propCNN, SGD with CNNprop, and SGD with CNNpropCNN. The first row is the all-in-focus image, which consists of all in-focus parts of the captured focal stack composited into a single image. PSNRs and SSIMs with respect to the target RGB image are reported. Second–fourth row: captured results focused at far, intermediate, and near distances.
Table 3. Comparison of different models on captured dataset. Top group: all trained on a single intensity plane (i.e., the intermediate plane) with the training set, PSNR evaluated on training and test sets. Center group: model trained on 7 of the 8 intensity planes (leave out the 1 held-out plane), PSNR evaluated on 1 (intermediate) plane for training and test set, respectively, held-out plane averaged over training and test sets on only that plane. Bottom group: all trained on 7 of the 8 intensity planes (leave out the 1 held-out plane), PSNR evaluated on these 7 planes for training and test set, respectively, held-out plane averaged over training and test sets on only that plane. Insights: (i) Among the single-plane models, our CNNpropCNN model is the best overall; (ii) a comparable version of our CNNpropCNN model trained on the 3D multiplane setting is almost as good as the single-plane model at that plane, implying that our model naturally extends to 3D; (iii) the performance at the held-out plane for the 3D model is best for variants of our model that use a CNN at the SLM plane; (iv) the CNNpropCNN variant of our model is the best.

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Set</th>
<th>Test Set</th>
<th>Held-out Plane</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASM</td>
<td>21.0</td>
<td>21.0</td>
<td>20.8</td>
</tr>
<tr>
<td>Chak. et al. [2020]</td>
<td>31.6</td>
<td>31.1</td>
<td>—</td>
</tr>
<tr>
<td>Peng et al. [2020]</td>
<td>31.8</td>
<td>31.4</td>
<td>25.2</td>
</tr>
<tr>
<td>Ours propCNN</td>
<td>32.6</td>
<td>32.4</td>
<td>25.7</td>
</tr>
<tr>
<td>Ours CNNprop</td>
<td>38.5</td>
<td>37.5</td>
<td>31.1</td>
</tr>
<tr>
<td>Ours CNNpropCNN</td>
<td><strong>40.0</strong></td>
<td><strong>38.8</strong></td>
<td>26.0</td>
</tr>
<tr>
<td>Tr. 7 Planes Ev. 1 Plane</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASM</td>
<td>21.0</td>
<td>21.0</td>
<td>20.8</td>
</tr>
<tr>
<td>Ours propCNN</td>
<td>32.1</td>
<td>32.0</td>
<td>29.3</td>
</tr>
<tr>
<td>Ours CNNprop</td>
<td>35.1</td>
<td>35.2</td>
<td>33.0</td>
</tr>
<tr>
<td>Ours CNNpropCNN</td>
<td><strong>37.9</strong></td>
<td><strong>37.9</strong></td>
<td><strong>33.1</strong></td>
</tr>
<tr>
<td>Tr. 7 Planes Ev. 7 Planes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASM</td>
<td>20.9</td>
<td>21.0</td>
<td>20.8</td>
</tr>
<tr>
<td>Ours propCNN</td>
<td>32.1</td>
<td>31.9</td>
<td>29.3</td>
</tr>
<tr>
<td>Ours CNNprop</td>
<td>35.2</td>
<td>34.9</td>
<td>33.0</td>
</tr>
<tr>
<td>Ours CNNpropCNN</td>
<td><strong>37.8</strong></td>
<td><strong>37.6</strong></td>
<td><strong>33.1</strong></td>
</tr>
</tbody>
</table>

Fig. S23. Frames extracted from the camera with different spatial shifts. The hologram is optimized using our CNNpropCNN model.
Fig. S24. Qualitative comparison of different models on captured test dataset. From left: Target image, ASM, HIL, NH, propCNN, CNNprop, CNNpropCNN. The model is trained on a single intensity plane with the training set, and the scenes visualized here are all from the test set (i.e. unseen data) at the same plane. Odd rows: Reconstruction results from unseen phases. Even rows: Absolute errors averaged over channels. The numbers on the right indicate the PSNR numbers in dB. Absolute errors are visualized in log scale and the colorbar is on the right.
Fig. S25. Qualitative comparison of different models on captured test dataset. From left: Target image, ASM, NH, propCNN trained on a single plane, CNNprop trained on a single plane, CNNpropCNN trained on a single plane, propCNN trained on 7 planes, CNNprop trained on 7 planes, CNNpropCNN trained on 7 planes. The scene visualized here is all from the test set (i.e. unseen data) at the same plane. The numbers on the left are propagation distances from the SLM in mm. Note that the fifth row (red box) is the plane supervised in a single-plane trained cases (From the first column to the fifth column). The sky box clearly shows that a CNN at the target plane does not generalize to other planes if only trained on a single plane. The third row is the held-out plane, which is not trained on even in multi-plane trained cases. The numbers on the top right indicates the PSNR numbers in dB.
Fig. S26. Qualitative comparison of different models on captured test dataset. From left: Target image, ASM, NH, propCNN trained on a single plane, CNNprop trained on a single plane, CNNpropCNN trained on a single plane, propCNN trained on 7 planes, CNNprop trained on 7 planes, CNNpropCNN trained on 7 planes. Here, the difference between the target is visualized for the scene in Fig. 25. The numbers on the left are propagation distances from the SLM in mm. The colormap indicates absolute errors averaged over three channels at each plane. Note that the fifth row (red box) is the plane supervised in a single-plane trained cases (From the first column to the fifth column). The sky box clearly shows that a CNN at the target plane does not generalize to other planes if only trained on a single plane. The third row is the held-out plane, which is not trained on even in multi-plane trained cases. Errors are visualized in log scale and the colorbar is on the top right.
Fig. S27. Qualitative comparison of different models on captured test dataset. From left: Target image, ASM, NH, propCNN trained on a single plane, CNNprop trained on a single plane, CNNpropCNN trained on a single plane, propCNN trained on 7 planes, CNNprop trained on 7 planes, CNNpropCNN trained on 7 planes. The scene visualized here is all from the test set (i.e. unseen data) at the same plane. The numbers on the left are propagation distances from the SLM in mm. Note that the fifth row (red box) is the plane supervised in a single-plane trained cases (From the first column to the fifth column). The sky box clearly shows that a CNN at the target plane does not generalize to other planes if only trained on a single plane. The third row is the held-out plane, which is not trained on even in multi-plane trained cases.
Fig. S28. Qualitative comparison of different models on captured test dataset. From left: Target image, ASM, NH, propCNN trained on a single plane, CNNprop trained on a single plane, CNNpropCNN trained on a single plane, propCNN trained on 7 planes, CNNprop trained on 7 planes, CNNpropCNN trained on 7 planes. Here, the difference between the target is visualized for the scene in Fig. 27. The numbers on the left are propagation distances from the SLM in mm. The colormap indicates absolute errors averaged over three channels at each plane. Note that the fifth row (red box) is the plane supervised in a single-plane trained cases (From the first column to the fifth column). The sky box clearly shows that a CNN at the target plane does not generalize to other planes if only trained on a single plane. The third row is the held-out plane, which is not trained on even in multi-plane trained cases. Errors are visualized in log scale and the colorbar is on the top right.
Fig. S29. Qualitative comparison of different models on captured test dataset. From left: Target image, ASM, NH, propCNN trained on a single plane, CNNprop trained on a single plane, CNNpropCNN trained on a single plane, propCNN trained on 7 planes, CNNprop trained on 7 planes, CNNpropCNN trained on 7 planes. The scene visualized here is all from the test set (i.e. unseen data) at the same plane. The numbers on the left are propagation distances from the SLM in mm. Note that the fifth row (red box) is the plane supervised in a single-plane trained cases (From the first column to the fifth column). The sky box clearly shows that a CNN at the target plane does not generalize to other planes if only trained on a single plane. The third row is the held-out plane, which is not trained on even in multi-plane trained cases.
Fig. S30. Qualitative comparison of different models on captured test dataset. From left: Target image, ASM, NH, propCNN trained on a single plane, CNNprop trained on a single plane, CNNpropCNN trained on a single plane, propCNN trained on 7 planes, CNNprop trained on 7 planes, CNNpropCNN trained on 7 planes. Here, the difference between the target is visualized for the scene in Fig. 29. The numbers on the left are propagation distances from the SLM in mm. The colormap indicates absolute errors averaged over three channels at each plane. Note that the fifth row (red box) is the plane supervised in a single-plane trained cases (From the first column to the fifth column). The sky box clearly shows that a CNN at the target plane does not generalize to other planes if only trained on a single plane. The third row is the held-out plane, which is not trained on even in multi-plane trained cases. Errors are visualized in log scale and the colorbar is on the top right.
Fig. S31. Qualitative comparison of different models on captured test dataset. From left: Target image, ASM, NH, propCNN trained on a single plane, CNNprop trained on a single plane, CNNpropCNN trained on a single plane, propCNN trained on 7 planes, CNNprop trained on 7 planes, CNNpropCNN trained on 7 planes. The scene visualized here is all from the test set (i.e. unseen data) at the same plane. The numbers on the left are propagation distances from the SLM in mm. Note that the fifth row (red box) is the plane supervised in a single-plane trained cases (From the first column to the fifth column). The sky box clearly shows that a CNN at the target plane does not generalize to other planes if only trained on a single plane. The third row is the held-out plane, which is not trained on even in multi-plane trained cases.
Fig. S32. Qualitative comparison of different models on captured test dataset. From left: Target image, ASM, NH, propCNN trained on a single plane, CNNprop trained on a single plane, CNNpropCNN trained on a single plane, propCNN trained on 7 planes, CNNprop trained on 7 planes, CNNpropCNN trained on 7 planes. Here, the difference between the target is visualized for the scene in Fig. S31. The numbers on the left are propagation distances from the SLM in mm. The colormap indicates absolute errors averaged over three channels at each plane. Note that the fifth row (red box) is the plane supervised in a single-plane trained cases (From the first column to the fifth column). The sky box clearly shows that a CNN at the target plane does not generalize to other planes if only trained on a single plane. The third row is the held-out plane, which is not trained on even in multi-plane trained cases. Errors are visualized in log scale and the colorbar is on the top right.
S6 ADDITIONAL RESULTS OF THE AR PROTOTYPE

In this section, we present additional experimental results for the AR prototype where a micro-prism-based lightguide is used to enable the visualization of both virtual holographic images and real scene images simultaneously. Note that we use only a green light source in this implementation, due to the lack of a second RGB laser module, and evaluate the display performance against that of baseline methods using a VR display mode in which the light incident from the real scene is blocked.

S6.1 Varifocal 2D Results in VR Mode (i.e., with room lights turned off)

We first show comparison results of displaying the USAF-1951 resolution chart at the equivalent farthest (3.9 m) and nearest (0.6 m) distances of our configuration (Figure S33). The holograms are generated using 3 different CGH algorithms, DPAC, SGD-ASM, and SGD-CNNpropCNN, respectively. We observe that the SGD-CNNpropCNN delivers the best image quality and highest light efficiency among all three investigated methods.

We then present in-focus and out-of-focus visualization of the same target image reconstructed at 3 equivalent depths (3.9 m, 1.2 m, 0.6 m) in Figure S34 and Figure S35. Holograms for these two sets are generated using SGD-ASM and SGD-CNNpropCNN, respectively. The red square highlighted patches indicate where the virtual target distance matches the lens focused distance. We observe that the SGD-CNNpropCNN delivers better image quality and higher light efficiency over SGD-ASM.

S6.2 3D Results in AR Mode

We present the full set of see-through results with displaying multiple virtual objects within the depth range of our configuration (Figure S36), as the complementary visualization to Section 5 of the main text. For the process of CGH, the input is a 2D target image and its corresponding depthmap, and the propagation model is implemented in a multiplane manner. We compare the DPAC algorithm, a multiplane SGD solver using the ASM model, and two variants of our multiplane 3D model. The first variant uses an SGD solver but only constrains in-focus scene amplitude, resulting in good image quality in those regions but noticeable out-of-focus speckle artifacts. The same model used with an ADMM solver that promotes piece-wise smooth phases for the in-focus parts of the scene exhibits good image quality for both in-focus and out-of-focus parts. Overall, we observe that algorithms using our wave propagation model perform better than those not using it. The SGD-CNNpropCNN method achieves the best in-focus results with ADMM-CNNpropCNN improving the out-of-focus content. The model is trained in the same manner as that of the VR prototype under the VR mode.
Fig. S33. Comparison of images reconstructed at 3 different depths (3.9 m, 1.2 m, 0.6 m) with 3 CGH algorithms, including DPAC, SGD-ASM, and SGD-CNNpropCNN. The exposure duration is set the same for fair comparison purposes. We note that DPAC suffers from essential ghosting copies that would significantly degrade the image quality and light efficiency.
Fig. S34. In-focus and out-of-focus visualization of images reconstructed at 3 depths with SGD-ASM. From top to bottom, virtual targets are placed at 3.9 m, 1.2 m, and 0.6 m. From left to right, the lens is focused at 3.9 m, 1.2 m, and 0.6 m. The red square highlighted patches indicate where the virtual target distance matches the lens focused distance.
Fig. S35. In-focus and out-of-focus visualization of images reconstructed at 3 depths with SGD-CNNDpropCNN. From top to bottom, virtual targets are placed at 3.9 m, 1.2 m, and 0.6 m. From left to right, the lens is focused at 3.9 m, 1.2 m, and 0.6 m. The red square highlighted patches indicate where the virtual target distance matches the lens focused distance.
Fig. S36. Comparison of 3D CGH methods using experimentally captured data with the lens focused at three different depths. We compare the DPAC algorithm (left), a multiplane SGD solver using the ASM model (center left), and two variants of our multiplane 3D model. The first variant uses an SGD solver but only constraints in-focus scene parts (center right). The second variant uses the same model but with an ADMM solver that promotes piecewise smooth phases for the in-focus parts of the scene (right). The three focused depths are around 3.9 m, 1.2 m, and 0.6 m, respectively. White arrows indicate virtual objects that are focused at a particular depth.
REFERENCES


