Joint Neural Phase Retrieval and Compression for Energy- and Computation-Efficient Holography on the Edge

YUJIE WANG*, Shandong University and Peking University, China

PRANEETH CHAKRAVARTHULA*, Princeton University and UNC Chapel Hill, USA

QI SUN, New York University, USA

BAOQUAN CHEN†, Peking University, China

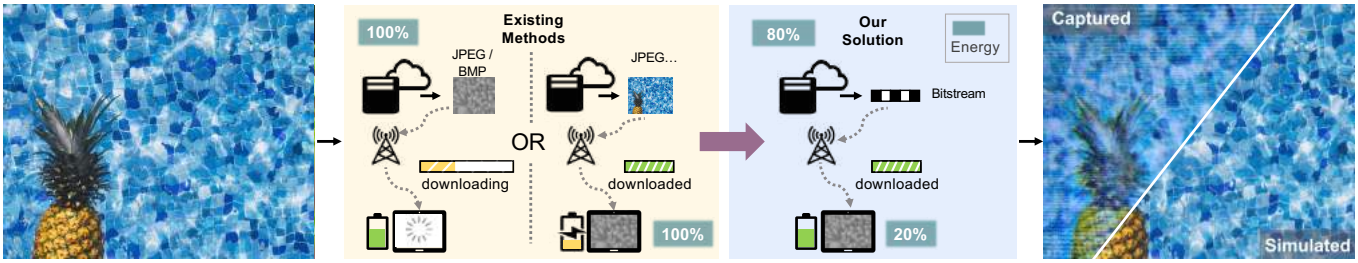

Fig. 1. To transmit a cloud-stored holographic image (a) and display on local devices, two natural solutions, one data transmission heavy, one local computation heavy, shown in (b) and (c) respectively, can be adopted. For (b), due to the statistical difference between holograms and natural images, high bit-rates are required for codecs (*e.g.*, JPEG) to ensure reconstruction quality, which may introduce high latency under bandwidth-limited Internet. On the other hand, for (c), shifting 100% of holographic phase retrieval computation to local reduces the latency, but inevitably elevates the energy costs on battery-constrained edge devices. To achieve the optimal latency/energy joint-performance, we propose a joint neural Phase Retrieval and Compression framework that partially shifts hologram computation to local devices while enabling transmission encoding with low bit-rates. (e) shows the simulated and captured display results.

Recent deep learning approaches have shown remarkable promise to enable high fidelity holographic displays. However, lightweight wearable display devices cannot afford the computation demand and energy consumption for hologram generation due to the limited onboard compute capability and battery life. On the other hand, if the computation is conducted entirely remotely on a cloud server, transmitting lossless hologram data is not only challenging but also result in prohibitively high latency and storage.

In this work, by distributing the computation and optimizing the transmission, we propose the first framework that jointly generates and compresses high-quality phase-only holograms. Specifically, our framework asymmetrically separates the hologram generation process into high-compute remote encoding (on the server), and low-compute decoding (on the edge) stages. Our encoding enables light weight latent space data, thus faster and efficient transmission to the edge device. With our framework, we observed a reduction of 76% computation and consequently 83% in energy cost on edge devices, compared to the existing hologram generation methods. Our framework is robust to transmission and decoding errors, and approach high image fidelity for as low as 2 bits-per-pixel, and further reduced average bit-rates and decoding time for holographic videos.

CCS Concepts: • **Hardware** → **Displays and imagers**.

Additional Key Words and Phrases: computer generated holography, neural hologram generation, hologram compression

ACM Reference Format:

Yujie Wang, Praneeth Chakravarthula, Qi Sun, and Baoquan Chen. 2022. Joint Neural Phase Retrieval and Compression for Energy- and Computation-Efficient Holography on the Edge. *ACM Trans. Graph.* 1, 1, Article 110 (July 2022), 15 pages.

*Equal contribution
†Corresponding author

Authors' addresses: Yujie Wang, Shandong University, Qingdao, and Peking University, Beijing, China, yujiew.cn@gmail.com; Praneeth Chakravarthula, Princeton University, Princeton, and UNC Chapel Hill, Chapel Hill, USA, cpk@cs.unc.edu; Qi Sun, New York University, Brooklyn, USA, qisun@nyu.edu; Baoquan Chen, School of AI, Peking University, Beijing, China, baoquan@pku.edu.cn.

1 INTRODUCTION

Cloud-based video streaming has revolutionized the means of media redistribution and consumption [Li et al. 2020]. Streaming services have spanned through consumer platforms such as mobile or virtual/augmented reality (VR/AR) devices (referred to as *edge* in this paper). On the other hand, holographic displays are a promising solution for future VR/AR, thanks to its low optical complexity and

high visual realism [Maimone et al. 2017]. For the future, we envision cloud-based holographic content streaming as an emerging demand, similar to the current 2D displays. However, the currently streamed media is commonly video sequences without a dedicated means supporting holograms.

Holographic displays utilize light diffraction to create a virtual 3D image after a medium with special patterns is lit, which is numerically computed or optically recorded for storing the light field information of 3D scenes and called *holograms*. When considering cloud based models for holographic displays, there are two direct solutions: (1) the server does the hologram generation and streams the phases to the edge, or (2) stream raw frames to the edge, which then generate holograms locally. However, for (1), subtle hologram compression loss may significantly harm the reconstruction quality since holograms are in phase domain [Jiao et al. 2018]. Ensuring full quality may cause long latency in interactive scenarios. For (2), edge devices are commonly unable to perform the demanding phase retrieval computation, thus introducing computation latency as well as extra battery consumption. For instance, a relatively less demanding face detection app runs out of battery within 40 minutes on a mobile AR device [LiKamWa et al. 2014]. Thus, it is essential to achieve both high-speed transmission to the edge and low-latency computation on the edge. As consumer devices are highly energy-constrained, this goal shall be generally realized without high computation load on the edge.

In this paper, we present a novel neural-network-based hologram generation and display framework that redistributes holographic generation computation between the cloud server and the edge, aiming to minimize both the necessary data transmission to the edge and the computation on the edge. The heart of our framework is a joint generation and compression of phase-only holograms, ensuring desirable computation-compression balance. Our approach asymmetrically separates the hologram generation process into high-computation encoding (on the server), and low-computation decoding (on the edge) stages. The encoding enables low latent space data, thus fast transmission to the edge. Under a cloud-based setting, the framework handles the computation which in turn lowers the energy consumption on future consumer-level holographic display devices. Consequently, we are able to achiever real-time high quality holographic display on the edge.

Our framework directly encodes input amplitude images and generates a compressed latent representation that can be decoded on the edge. We adopt a hyper prior model from [Ballé et al. 2018] that extracts side information to model the distribution of the encoded latent representation and a straight-through gradient estimator [Bengio et al. 2013] to back-propagate the gradients from the non-differentiable rounding operation. Instead of individual pixels, we encode/decode the hologram from the deep latent spaces. In a run-time cloud-edge system, only the highly compressed latent vectors are transmitted through the network.

A series of experiments with simulated cloud-edge frameworks demonstrate our significant advantage on low data transmission (18× compression), low local computation cost, thus high energy efficiency (about 20%) , and robust-to-noise. Specifically, the reconstruction quality remains similar when the compressed latent data is contaminated by noise sampled from zero-mean normal distribution with σ between 0.01 and 0.5. The proposed framework enables decoding on client side at around 30 frames per second (single channel). In summary, we present an end-to-end system that jointly optimizes holographic transmission and computation for future cloud-based platforms. Codes and data for this paper are available at this link[1]. The presented research makes the following major contributions:

- A novel scheme for coupling hologram generation and compression to reduce transmission latency from the remote cloud servers.
- A neural framework with asymmetric distribution of compute between remote servers and edge devices, which reduces computational and energy cost at the edge devices, and achieves efficient decoding.
- Extensive evaluation of the framework's effectiveness in simulation and on an experimental hardware prototype, and exhaustive assessment of the proposed framework's robustness to noise and scalability to holographic videos.

2 RELATED WORK

2.1 Computational Holography

Computer generated holography (CGH) numerically simulates the complex optical wave propagation process from virtual objects. It has the potential to reproduce focus [Choi et al. 2021; Shi et al. 2021] and parallax cues [Chakravarthula et al. 2022], and also correct for aberrations in the eye [Chakravarthula et al. 2021; Maimone et al. 2017]. A spatial light modulator (SLM) modulates the wavefront of incident light in a holographic display. Existing SLMs unfortunately cannot modulate both amplitude and phase, and hence a phase-only SLM is typically used for its higher diffraction efficiency. However, this requires generating a phase-only hologram that can produce the desired image intensity after propagation, which is indeed the core challenge of computer generated holography.

Representing the target scene as a collection of point light sources or polygonal meshes with individual emitters is a widely used representation [Benton and Bove Jr 2008; Ogihara and Sakamoto 2015]. Point based methods treat each point in a point cloud as a spherical light source and compute the corresponding interference pattern at the hologram plane to generate the final hologram. On the other hand, due to the popularity of polygon representation in computer graphics pipelines, polygon based methods [Kim et al. 2008; Matsushima 2005] often utilize Fast Fourier Transform (FFT) along with an additional coordinate transformation to calculate the diffraction patterns from tilted and shifted polygonal planes. However, both methods demand heavy compute as they require a dense set of primitives for representing a given scene. For enhancing the computation efficiency, various optimization techniques are proposed, such as GPU parallelization [Chen and Wilkinson 2009; Masuda et al. 2006; Petz and Magnor 2003], look-up tables [Kim and Kim 2008] with intermediate wavefront recording planes [Shimobaba

[1] https://github.com/HoloCompress/DPRC

et al. 2009]. However, physically based methods typically face challenges in reproducing view-dependent effects [Zhang et al. 2017] in 3D scenes.

Meanwhile, image-based approaches generally offer better computational efficiency and are favored for modeling occlusions and other view-dependent effects [Chakravarthula et al. 2022; Padmanaban et al. 2019]. Two popular image-based hologram approaches are lightfield holograms and layer-based multifocal methods. Both methods render a 3D scene either as a set of lightfield images from multiple view points or a stack of images at multiple focal planes. Calculation of holograms then is done by accumulating the wavefronts propagated from the image-based representation of the 3D scene to the hologram plane.

In the past several years, great success is achieved by deep neural networks in solving some of the difficult problems in computer vision and computer graphics. Recently, researchers have started applying neural networks for solving the holographic phase retrieval problem, and a few application-specific CGH methods have been proposed. For instance, neural networks have been applied to holography [Choi et al. 2021; Eybposh et al. 2020; Peng et al. 2020; Rivenson et al. 2018; Shi et al. 2021], ptychography [Boominathan et al. 2018], coherent diffraction imaging (CDI) [Cherukara et al. 2018; Goy et al. 2018], and quantitative phase microscopy [Kellman et al. 2019; Kemp 2018]. Image quality of holographic displays were further improved by optimizing holograms in a hardware-in-the-loop fashion [Chakravarthula et al. 2020a; Peng et al. 2020]. In this work, we propose the first method devised for cloud-based consumer holographic displays by jointly optimizing the image quality, compute and data transmission.

2.2 Image Compression

Traditional image compression codecs, such as JPEG [Wallace 1992] and JPEG2000 [Taubman and Marcellin 2013], consist of multiple modules including transformations, quantization and entropy coding. In the JPEG compression standard, Discrete Cosine Transform (DCT) is applied to each 8×8 pixel patch extracted from the input image, after which original information is transformed into decorrelated coefficients. Quantization is then applied to discard less significant information by truncating the coefficient vectors. Entropy coding is then used for lossless encoding of the information. However, the individual modules of traditional image compression codecs are difficult to optimize jointly [Hu et al. 2021], thus limiting the compression performance [Ma et al. 2020].

Recently, deep-learning-based models are extensively leveraged to perform compression [Ballé et al. 2018; Ballé et al. 2017; Mentzer et al. 2020; Minnen et al. 2018]. Balle *et al.* [2017] proposed a CNN based end-to-end image compression method. However, the performance of their fully factorized entropy model unfortunately degraded with statistical dependencies in latent representations. A hyperprior model proposed in [Ballé et al. 2018] reduced the data redundancy by exploiting the spatial dependencies. Minnen *et al.* [2018] adopted an auto-regressive prior information to further mitigate the data redundancy. These models are effective yet slow as the pixels are decoded sequentially, making them less applicable for high-resolution images. More recent work by Mentzer *et al.* [2020] utilizes Generative Adversarial Network (GAN) to achieve appealing reconstruction quality with considerably low bit-rates. Such image compression techniques cannot be directly applied for phase hologram data as we demonstrate in this work. Moreover, end-to-end compression pipelines have not been realized so far for hologram data, which we believe will soon become important for consumer holographic displays and holographic storage.

3 COMPUTER GENERATED HOLOGRAPHY

Computer generated holography (CGH) numerically simulates the optical process of hologram recording and replay [Chakravarthula et al. 2019]. A phase-only spatial light modulator (SLM) is typically used in a holographic display for its light efficiency. However, the calculation of the phase pattern that results in an intended intensity image is often challenging and computationally expensive. In this section, we briefly discuss holographic phase retrieval.

In a holographic display, as shown in Figure 2(c), the phase hologram $\mathbf{H}$ displayed on an SLM modulates the phase of an incident coherent beam U_s, which propagates over a distance d in free space to produce an interference pattern, whose intensity is the intended target image. Such interference pattern can be calculated by Rayleigh-Sommerfeld (RS) scalar diffraction integral, given by

$$f_p^d(U_s, \mathbf{H})\big|_{(x,y)} = \frac{1}{j\lambda} \iint_{\Sigma} U_s(\xi,\eta) e^{j\mathbf{H}(\xi,\eta)} \frac{\exp(jkr)}{r} d\xi d\eta, \quad (1)$$

where λ is the wave length, $k = \frac{2\pi}{\lambda}$ is the wave number and Σ represents the aperture of the hologram plane. $U_s(\xi,\eta)e^{j\mathbf{H}(\xi,\eta)}$ is the field at the hologram plane and $r = \sqrt{(\xi-x)^2 + (\eta-y)^2 + d^2}$ is the Euclidean distance between any point (ξ,η) on the hologram plane and any pixel (x,y) on the image plane.

While the above integral gives perhaps the most accurate scalar diffraction field, it is computationally very expensive. Therefore, various simplifying assumptions have been made to efficiently compute the RS integral. Herein, we adopt the band-limited angular spectrum propagation model [Matsushima and Shimobaba 2009]:

$$f_p^d(U_s, \mathbf{H})\big|_{(x,y)} = \iint \mathcal{F}(U_s e^{j\mathbf{H}})\big|_{(\xi,\eta)} \mathbb{H}(u_\xi, u_\eta) e^{j2\pi(u_\xi \xi + u_\eta \eta)} du_\xi du_\eta,$$
$$\mathbb{H}(u_\xi, u_\eta) = \begin{cases} e^{j2\pi d\sqrt{\frac{1}{\lambda^2} - u_\xi^2 - u_\eta^2}}, & \text{if } \sqrt{u_\xi^2 + u_\eta^2} < \frac{1}{\lambda}, \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

where u_ξ, u_η are the spatial frequencies and $\mathcal{F}(\cdot)$ represents the Fourier transform. Since only the intensity of the wave field is observed by human eyes (or cameras), the observed image is given by $|\widehat{\mathbf{A}}_t|^2 = |f_p^d(U_s, \mathbf{H})|^2$, where $|\cdot|$ denotes the element-wise absolute value. The holographic phase retrieval problem aims at finding a phase pattern that matches the resulting intensity $|\widehat{\mathbf{A}}_t|^2$ match a given target intensity $|\mathbf{A}_t|^2$. In other words, holographic phase retrieval solves the following optimization problem:

$$\mathbf{H} = \arg\min_{\mathbf{H}} \mathcal{L}(|\widehat{\mathbf{A}}_t|^2, |\mathbf{A}_t|^2), \quad (3)$$

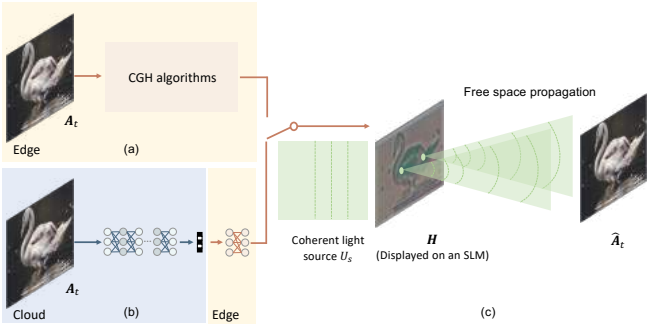

Fig. 2. A simplified illustration for the cloud-edge collaborative holographic displaying model. Prevalent solutions mainly adopt the strategy shown in (a), where the phase data is computed at the edge side, while the proposed framework enables a collaborative cloud-edge displaying solution, as shown in (b). (c) shows a simplified holographic display model. When illuminated by the coherent light source U_s, the spatial light modulator (SLM) modulates the phase of the light wave according to the hologram $\mathbf{H}$. The modulated light wave arrives at the target plane after propagating in space. The content perceived by human eyes (or captured by cameras) is mainly from the intensity of the complex wave field, *i.e.*, mainly from the amplitude $\hat{\mathbf{A}}_t$.

where $\mathcal{L}$ denotes a custom penalty function.

Computing a phase-only hologram on a local device as illustrated in Figure 2(a)+(c) requires computation of high-quality phase patterns $\mathbf{H}$, such that the optically reconstructed image $\widehat{\mathbf{A}}_t$ is close to the given target image $\mathbf{A}_t$. However, this demands a significant compute and power, and thus it is desirable to instead perform the phase computation remotely and transmit to the edge device, especially in case of wearable near-eye displays that are expected to work all day long. To remove the latency bottleneck and maintain the quality of phase transmission, we propose a learning-based joint phase retrieval and compression framework to specifically tailor for lightweight cloud-edge devices as illustrated in Figure 2(b)+(c). We describe our joint phase retrieval and compression framework in the following section.

4 JOINT NEURAL PHASE RETRIEVAL AND COMPRESSION

In this section, we describe our joint hologram phase generation and compression framework to achieve phase-only holograms that use significantly lower bits per pixel compared to the state of the art CGH methods, but result in holographic images that are on par with the existing optimization-based and neural network-based methods. Specifically, we introduce a learned feature encoding and real-time data decoding framework as illustrated in Section 4.1. Our framework achieves significantly lower transmission data volumes (from the cloud), and low computational cost (in GFLOPs, on the edge), without compromising the quality of reconstructed holographic images. In Section 4.2, we discuss in detail the latent code compression and bit quantization scheme. In Section 4.3, we extend the proposed framework to exploit the redundancies in consecutive video frames for achieving higher transmission efficiency on holographic videos.

To achieve low compute and high quality reconstructions on the edge, we asymmetrically distribute the hologram generation between the cloud and edge devices. Specifically, the cloud servers which have stronger computational resources generate a *latent space compressed* reduced volume transmission data, which can be decoded on the edge device at significantly lower compute and energy cost. We illustrate our framework for joint phase retrieval and compression in Figure 3 and all the involved notations are summarized in Table 1 for clarity. Also note that the phase retrieval network (PRN) modules comprise of $\{IP, E_p, D_p\}$, the coding related modules include a hyper-prior encoder/decoder $\{E_h, D_h\}$ and the differentiable quantizers include $\{Q_s, Q_n\}$, as illustrated in Figure 3.

Table 1. Table of variables

Symbol	Data Type	Dimension	Description
$\mathbf{A}_t$	Float	$H \times W \times 1$	Target amplitude map
$\mathbf{P}_t$	Float	$H \times W \times 1$	Output from IP
$\mathbf{H}$	Float	$H \times W \times 1$	Generated hologram $\mathbf{H}$
$\mathbf{v}$	Float	$\frac{H}{4} \times \frac{W}{4} \times 8$	Latent space
$\mathbf{z}$	Float	$\frac{H}{16} \times \frac{W}{16} \times 64$	Hyper-latent
$\boldsymbol{\mu}$	Float	$\frac{H}{4} \times \frac{W}{4} \times 8$	Mean of the Gaussian model for $\mathbf{v}$
$\boldsymbol{\sigma}$	Float	$\frac{H}{4} \times \frac{W}{4} \times 8$	Scale of the Gaussian model for $\mathbf{v}$
$\widehat{\mathbf{A}}_t$	Float	$H \times W \times 1$	Simulated reconstruction of $\mathbf{A}_t$
$\hat{\mathbf{v}}$	Integer	$\frac{H}{4} \times \frac{W}{4} \times 8$	Quantized $\mathbf{v}$
$\hat{\mathbf{z}}$	Integer	$\frac{H}{16} \times \frac{W}{16} \times 64$	Quantized $\mathbf{z}$
$c_{\mathbf{v}}$	Binary bits	-	Bitstream coded for $\hat{\mathbf{v}}$
$c_{\mathbf{z}}$	Binary bits	-	Bitstream coded for $\hat{\mathbf{z}}$

-: The lengths of the bitstreams are dynamically changed according to the propability distribution of the elements within the data.

4.1 Holographic Phase Retrieval

The overall phase retrieval module of our framework is illustrated in Figure 3. Along with the phase retrieval as described in Equation (3), our phase retrieval network (PRN) also includes a feature encoding E_p and a phase decoding D_p step. In the feature encoding step, a target amplitude $\mathbf{A}_t$ is combined with a neural-network initialized phase map $\mathbf{P}_t$, predicted by the Initial Phase predictor (IP) sub-network, to form a complex wave field that is numerically propagated to the SLM plane. Then, latent features $\mathbf{v}$ of the propagated holographic field $\{\mathbf{A}_s, \mathbf{P}_s\}$ are encoded for compression and transmission to the edge. In the decoding stage, the hologram $\mathbf{H}$ is generated from the transmitted compressed features $\mathbf{v}$. We now discuss these two processes in detail hereunder.

Feature Encoding. Given a target image amplitude, we initialize the unknown target image phase $\mathbf{P}_t$ as predicted by the sub-network IP (initial phase predictor), as shown in Figure 3. The complex-valued wave field $\{\mathbf{A}_t, \mathbf{P}_t\}$ at target plane is then numerically propagated to the SLM plane, formulated by

$$\{\mathbf{A}_s, \mathbf{P}_s\} = f_p^d(\mathbf{A}_t, \mathbf{P}_t), \tag{4}$$

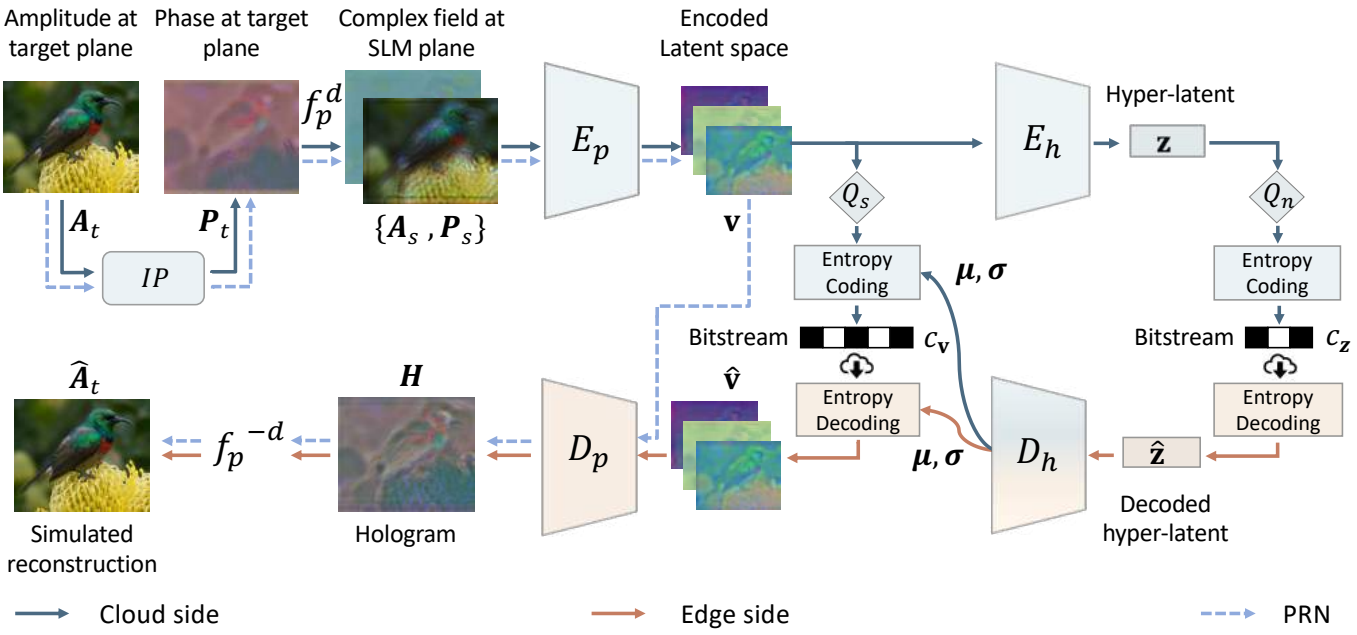

Fig. 3. Workflow of the proposed cloud-based holography framework. The modules within the framework are deployed separately on the cloud side and the edge side. On the cloud side, a feature extraction and coding pipeline is performed. After obtaining the target amplitude map $\mathbf{A}_t$, the initial phase predictor *IP* predicts an initial phase distribution $\mathbf{P}_t$ at the target plane. A complex wavefield $\{\mathbf{A}_t, \mathbf{P}_t\}$ is formed and propagated to the SLM plane via a simulated propagation process f_p^d. Then, the propagated wavefield $\{\mathbf{A}_s, \mathbf{P}_s\}$ undergoes feature extraction performed by E_p, and the resulting latent vector $\mathbf{v}$ is coded to the bitstream $c_\mathbf{v}$. To efficiently code $\mathbf{v}$, its data distribution is modelled by a hyper encoder-decoder $\{E_h, D_h\}$, where a hyper-latent $\mathbf{z}$ is introduced and coded into another bitstream $c_\mathbf{z}$. On the edge side, the final phase map $\mathbf{H}$ is generated from the decoded $\hat{\mathbf{v}}$ based on the probability distribution predicted from $\hat{\mathbf{z}}$. Since the same probability distribution is required for entropy coding and decoding, D_h is duplicated and deployed on the cloud side and each edge side. PRN is an extracted sub-network that performs phase retrieval only. The grayscale phases for R,G,B channels are arranged in 3-channel color images for visualization.

where $\mathbf{A}_s$ and $\mathbf{P}_s$ denote the amplitude and phase at the SLM plane respectively, and $f_p^d(\cdot,\cdot)$ represents the band-limited angular spectrum (AS) propagation method as described in Equation (2). We now use a feature encoder E_p to encode the complex wave field at the SLM plane $\{\mathbf{A}_s, \mathbf{P}_s\}$ to a latent space $\mathbf{v}$. Specifically, we use a multi-scale structural encoder for fully encoding the information contained in the complex field $\{\mathbf{A}_s, \mathbf{P}_s\}$. We show a more detailed illustration of the encoder in Figure 4. As can be seen, we adopt several parallel branches in E_p to extract features from $\{\mathbf{A}_s, \mathbf{P}_s\}$ at different scales before producing the latent features $\mathbf{v}$.

Phase Decoding. The SLM complex field features will be compressed to significantly reduce the data volume (Section 4.2). The features $\mathbf{v}$ are used by the decoder D_p to recover the phase hologram $\mathbf{H}$. For the decoder sub-network, we employ a residual architecture containing m residual blocks. Adopting residual blocks contributes to an efficient feature flow and gradient flow during the backward propagation. We formulate decoding the phase hologram $\mathbf{H}$ from the latent SLM field features $\mathbf{v}$ as follows:

$$\mathbf{H} = D_p(\mathbf{v}). \tag{5}$$

From the recovered phase hologram $\mathbf{H}$, the reconstructed image amplitude is computed as

$$\widehat{\mathbf{A}}_t = |f_p^{-d}(\mathbf{1}, \mathbf{H})|, \tag{6}$$

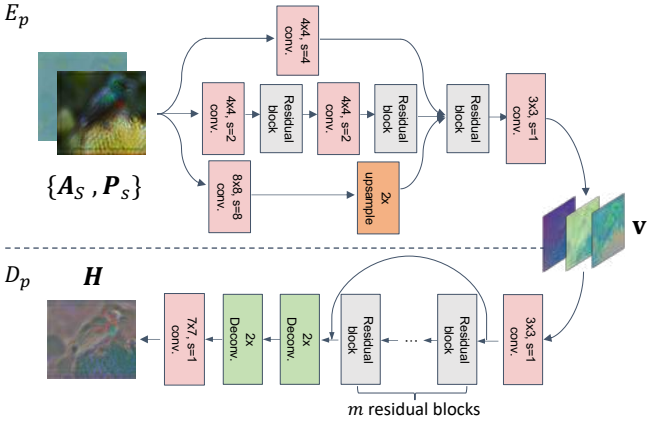

Fig. 4. Multi-scale encoder E_P and residual decoder D_p. Within the encoder E_p, multiple branches are designed for enabling better feature extraction and more efficient backward gradient flow.

where $f_p^{-d}(\cdot,\cdot)$ denotes the backward wave propagation from the SLM plane to the image plane.

Phase Retrieval Penalty Functions. The phase retrieval network (PRN) is trained to minimize the reconstruction error $\mathcal{L}_r$ between the reconstructed image $\widehat{\mathbf{A}}_t$ and the target image $\mathbf{A}_t$. Note that as $\widehat{\mathbf{A}}_t$

is reconstructed from the hologram $\mathbf{H}$, supervision on $\widehat{\mathbf{A}}_t$ imposes constraints on $\mathbf{H}$ as well. We use Mean Squared Error (MSE) as a per-pixel penalty and Muti-scale Structural Similarity (MS-SSIM) [Wang et al. 2003] as a perceptual metric. We use learned perceptual error metrics including LPIPS-VGG [Zhang et al. 2018] and Watson-DFT [Czolbe et al. 2020] to further improve the reconstruction quality for a human observer. Therefore, the overall optimization penalty function is calculated as follows:

$$\mathcal{L}_r = \mathcal{L}_{\text{mse}} + \alpha_{\text{mss}}\mathcal{L}_{\text{mss}} + \alpha_{\text{vgg}}\mathcal{L}_{\text{vgg}} + \alpha_{\text{wdft}}\mathcal{L}_{\text{wdft}}, \tag{7}$$

where $\mathcal{L}_{\text{mse}}$, $\mathcal{L}_{\text{mss}}$, $\mathcal{L}_{\text{vgg}}$ and $\mathcal{L}_{\text{wdft}}$ denote MSE, MS-SSIM loss, LPIPS-VGG loss and Watson-DFT loss respectively, and α_{mss}, α_{vgg}, and α_{wdft} denote the corresponding balance weights. The VGG network-based loss $\mathcal{L}_{\text{vgg}}$ is calculated as

$$\mathcal{L}_{\text{vgg}} = \sum_l w_l ||\phi_l(\widehat{\mathbf{A}}_t) - \phi_l(\mathbf{A}_t)||_2^2, \tag{8}$$

where ϕ_l represents l_{th} layer of a pre-trained VGG-19 [Simonyan and Zisserman 2015] network. $\mathcal{L}_{\text{vgg}}$ is adopted to achieve finer details in the reconstructed image by penalizing the features at multiple layers from the VGG-19 network. However, as stated in [Czolbe et al. 2020], a pre-trained network optimized for classification task tends to underestimate the perceptual influence of graphical artifacts such as noise. Besides, [Czolbe et al. 2020] demonstrates that a generation network optimized using LPIPS-VGG loss might introduce noticeable artifacts in the reconstruction results. As Watson-DFT function proposed in [Czolbe et al. 2020] is more sensitive to frequency changes, as would be for a human observer, we adopt it to further improve the reconstruction quality.

4.2 Latent Compression

The usage of encoder and decoder sub-networks within the phase retrieval stage, as discussed above in Section 4.1, already reduces the number of elements to be processed to about half. Although the latent space $\mathbf{v}$ is about half in size compared to the target $\mathbf{A}_t$, the data volume needed for transmitting the floating point values of the SLM field features $\mathbf{v}$ from the remote server to the edge device is still very large. For example, storing the features $\mathbf{v}$ of a single-channel phase hologram $\mathbf{H}$ with resolution 1080×1920 costs about 32 MB in space. This necessitates a compression framework for lightweight storage, transmission and processing.

As a high data precision demands a high bit-rate to encode information, the latent space $\mathbf{v}$ needs to be quantized before being coded into binary bits, so that the elements become more discretized and require less bits. Therefore, as shown in Figure 3, a quantizer Q_s is introduced to quantize $\mathbf{v}$ to $\hat{\mathbf{v}}$. Simultaneously, to utilize entropy coding methods for achieving efficient coding, the data distribution of the elements in $\hat{\mathbf{v}}$ needs to be modelled. Since the actual marginal distribution of $P_{\hat{\mathbf{v}}|\mathbf{A}_t}$ of $\hat{\mathbf{v}}$ is unknown, a hyperprior network proposed in [Ballé et al. 2018], formed by $\{E_h, D_h\}$, is equipped to model the data distribution as an entropy model $p_{\hat{\mathbf{v}}}$. To make the full framework able to be optimized in an end-to-end manner, the bit-rate needs to be effectively measured or estimated in a differentiable manner and the differentiable alternatives for the real rounding operations are incorporated. Besides, different from learning based compression methods [Ballé et al. 2018; Mentzer et al. 2020; Minnen et al. 2018] that pursue an exact reconstruction of the input to the feature encoder, we utilize D_p to directly generate a different output, *i.e.*, a phase-only hologram $\mathbf{H}$, from the transmitted $\hat{\mathbf{v}}$. We annotate the entire framework ***Dual Phase Retrieval and Compression (DPRC)***. The details of each module are described below.

Quantization. For using finite bits to encode data losslessly, discretization is needed to make the symbols coming from a discrete set [Gray 2011]. Rounding is a commonly used discretization technique. However, a real rounding operation is not differentiable. Inspired by [Theis et al. 2017], we adopt a differentiable alternative Q_s, which is defined as

$$\begin{aligned} \hat{\mathbf{v}} &= Q_s(\mathbf{v}) \\ &= sg([\mathbf{v}] - \mathbf{v}) + \mathbf{v}, \end{aligned} \tag{9}$$

where Q_s denotes the quantizer with stop-gradient operation $sg(\cdot)$ that blocks gradients flowing into its argument and $[\cdot]$ represents rounding operation. By using Q_s, the rounding operation is exerted as usual in both training and test process, and gradients of $\hat{\mathbf{v}}$ directly flow to E_p, which means the rounding operation is bypassed in back-propagation. Although there are other smooth approximations for rounding, adopting Q_s is helpful to resolve the mismatch problem introduced by using smooth rounding approximations for training but using real rounding for inference stage.

Bit-rate Estimation. Since $\hat{\mathbf{v}}$ is discretized, it can be coded losslessly by introducing a probability model $P_{\hat{\mathbf{v}}}$ of $\hat{\mathbf{v}}$ and using an entropy coding method such as arithmetic coding [Rissanen and Langdon 1981]. According to Shannons rate-distortion theory [Cover and Thomas 2006], the bit-rate for coding $\hat{\mathbf{v}}$ by the entropy model $P_{\hat{\mathbf{v}}}$ is lower-bounded by

$$R_{\hat{\mathbf{v}}} = \mathbb{E}_{\hat{\mathbf{v}}\sim P_{\hat{\mathbf{v}}|\mathbf{A}_t}}[-log_2 P_{\hat{\mathbf{v}}}(\hat{\mathbf{v}})]. \tag{10}$$

If $\hat{\mathbf{v}}$ is perfectly coded, *i.e.*, the entropy model $P_{\hat{\mathbf{v}}}$ exactly matches the actual marginal distribution of $\hat{\mathbf{v}}$ (the unknown distribution $P_{\hat{\mathbf{v}}|\mathbf{A}_t}$), the bit-rate is minimized. For modeling $P_{\hat{\mathbf{v}}}$, we choose a conditional Gaussian model adopted in [Minnen et al. 2018] for capturing the spatial dependencies within $\hat{\mathbf{v}}$, given by

$$P_{\hat{v}|\hat{z}} \sim \mathcal{N}(\boldsymbol{\mu}, \text{diag}(\boldsymbol{\sigma})). \tag{11}$$

$\boldsymbol{\mu}$, $\boldsymbol{\sigma}$ are mean and scale for the Gaussian model, which are estimated by a hyperprior sub-network denoted as E_h and D_h in Figure 3. $\hat{\mathbf{z}}$ is a quantized hyper-latent representation that is encoded by E_h from $\mathbf{v}$. The hyper-latent $\hat{\mathbf{z}}$ is introduced to capture the spatial dependencies within $\mathbf{v}$ and make the elements in $\mathbf{v}$ conditionally independent. Different from [Mentzer et al. 2020], which utilizes two separate sub-networks for predicting $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$, we use a smaller sub-network D_h for predicting both $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ to reduce the computational cost of the decoding process. Although the auto-regressive decoding procedure proposed in [Minnen et al. 2018] shows better performance, it is inefficient as it sequentially decodes the pixels. Considering the time efficiency, we adopt the decoding method used in [Ballé et al. 2018; Ballé et al. 2017], by which all of the elements are recovered in parallel via convolutional layers. As $\hat{\mathbf{z}}$ is needed for predicting

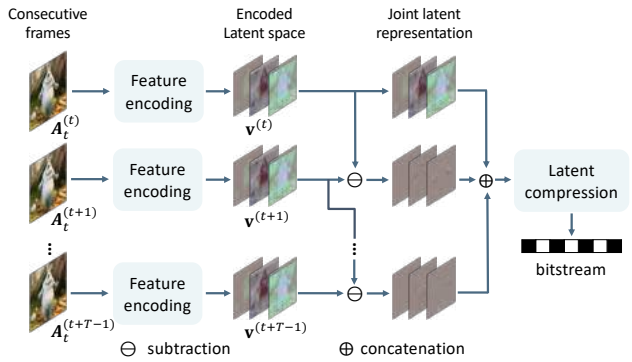

Fig. 5. Process for jointly compressing consecutive holographic video frames. The framework receives T frames and encodes the frames into a set of latent space using the same feature encoding process. Then a joint latent representation is constructed and fed into the latent compression modules to obtain the bitstream for transmission.

the parameters of the Gaussian model for decoding $\hat{\mathbf{v}}$ from the bitstream $c_{\mathbf{v}}$, it is coded to a another bitstream $c_{\mathbf{z}}$ and transmitted. Then the bit-rate consumed by $c_{\mathbf{z}}$ is estimated by

$$\begin{aligned} R_{\hat{\mathbf{z}}} &= \mathbb{E}[-log_2 P_{\hat{\mathbf{z}}}(\hat{\mathbf{z}})] \\ &= \mathbb{E}[-log_2 P_{\hat{\mathbf{z}}|\hat{\mathbf{v}}}(Q_n(E_h(\hat{\mathbf{v}})))], \end{aligned} \tag{12}$$

where Q_n denotes a simulated quantization by additive i.i.d. uniform noise as it has the same width as the quantization bins (one) [Ballé et al. 2016]. $P_{\hat{\mathbf{z}}}$ is modeled by a factorized entropy model [Ballé et al. 2017], for which the detailed derivation are provided in the supplementary material.

Rate-Generation Loss. After the compression modules are introduced, the DPRC framework is trained with the loss given in Equation (13), which exhibits a trade-off between hologram generation quality and the bit cost.

$$\begin{aligned} \mathcal{L}_c &= R + \alpha_r \mathcal{L}_r \\ &= (R_{\hat{\mathbf{v}}} + R_{\hat{\mathbf{z}}}) + \alpha_r \mathcal{L}_r. \end{aligned} \tag{13}$$

where α_r is used to adjust the role of the reconstruction loss term to achieve different quality levels of the produced holograms with various degrees of data volume reduction.

4.3 Redundancy-based Holographic Video Compression

Considering the prevalence of video transmission [Li et al. 2020] and similarities existed in consecutive frames, we design a prototype for compressing holographic video frames to further reduce the average volume of each frame. As shown in Figure 5, the framework compresses T frames jointly using a conditional entropy model on top of the latent representation $\{\mathbf{v}^{(i)}\}_{i=t}^{t+T-1}$ generated for T frames in parallel. Since the latent representation $\mathbf{v}^{(i)}$ generated from a complex field is highly different from natural images, optical flow-based transformation commonly adopted in natural video compression becomes less feasible. This is due to the difficulty of accurately predicting the optical flow between every two adjacent elements in $\{\mathbf{v}^{(i)}\}_{i=t}^{t+T-1}$, which usually have no obvious semantic structures. Additionally, as utilizing optical flows will require extra bits to store flow maps, we choose to construct a joint latent representation $\mathbf{v}_T$ from $\{\mathbf{v}^{(i)}\}_{i=t}^{t+T-1}$ without flow based transformation. Specifically, $\mathbf{v}_T$ is constructed by

$$\mathbf{v}_T = \oplus\{\mathbf{v}^{(t)}, \mathbf{v}^{(t+1)} - \mathbf{v}^{(t)}, ..., \mathbf{v}^{(t+T-1)} - \mathbf{v}^{(t+T-2)}\}, \tag{14}$$

where $\oplus$ denotes concatenation. Equation (14) shows that $\mathbf{v}_T$ contains the untouched latent space $\mathbf{v}_t$ and the residuals between every two frames with indices in $[t, t+T-1]$. Storing residuals for $\{\mathbf{v}^{(i)}\}_{i=t+1}^{t+T-1}$ is beneficial for further reducing the data volume since there are usually subtle differences between consecutive frames. $\mathbf{v}_T$ then undergoes the latent compression procedure given in Section 4.2. Specifically, we quantize $\mathbf{v}_T$ to $\hat{\mathbf{v}}_T$ and predict the parameters of the probability model for elements within $\hat{\mathbf{v}}_T$. Later, $\hat{\mathbf{v}}_T$ is coded by the entropy coding module utilizing the predicted probability models. During the decoding stage, the latent representations for each frame are sequentially recovered and fed into the phase decoder D_p to generate corresponding holograms.

5 IMPLEMENTATION

Here, we discuss the implementation of our DPRC framework and the prototype display used for experimental evaluation. Please refer to the Supplementary Material for additional details and a detailed discussion.

DPRC Framework. We implemented the entire DPRC framework in PyTorch [Paszke et al. 2019], with the neural network trained in two stages, and on 800 images from the DIV2K dataset [Timofte et al. 2017]. Specifically, the sub-network for phase retrieval (PRN) is trained with the penalty function defined in Equation (7) in the first stage, and the full pipeline is trained using the rate-generation loss as described in Equation (13) in the second stage. During training, we adopt a rate constraining strategy [Mentzer et al. 2020] to avoid any drastic reduction in bit-rate. The entropy coding/decoding is implemented based on the rANS (Range Asymmetric Numeral System) [Duda 2014] coder provided by CompressAI library [Bégaint et al. 2020].

Prototype Display. Our hardware prototype used a HOLOEYE Leto LCoS reflective SLM with a pixel pitch of 6.4μm and 1080×1920 pixel resolution. We use a 4F relay system with an aperture at the Fourier plane to filter any higher diffraction orders arising from the double phase encoded holograms. The virtual SLM after the 4F system relays the images directly onto the camera sensor for measurements. We use two Pentax 645n 75mm lenses for constructing our 4F system and a Canon Rebel t6i camera sensor body (without the lens attached) for measuring the displayed images for quality assessment. The camera has an output resolution of 6000 Œ 4000 and a pixel pitch of 3.72 μm, well above the pitch of our SLM. The SLM is controlled as an external monitor and the hologram phase patterns are transferred and displayed on it via the HDMI port of the graphics card. This SLM is illuminated by a collimated and linearly polarized beam from a single optical fiber that is coupled to three laser diodes. The laser diodes emit at wavelengths 450 nm, 520 nm and 638 nm and are controlled in a color field sequential manner.

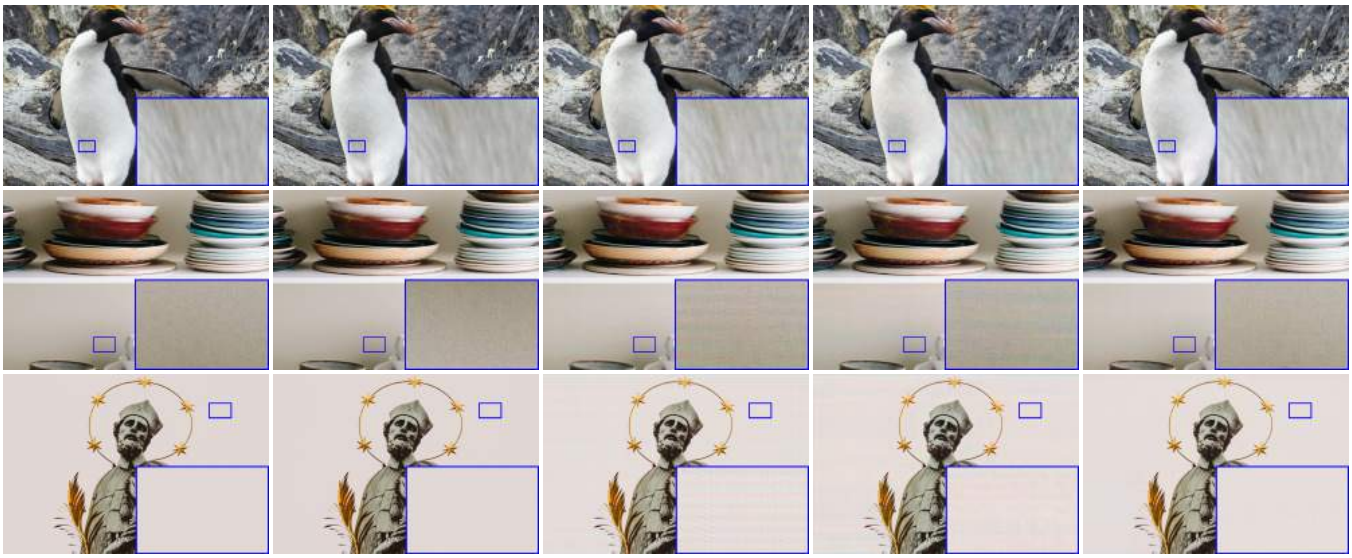

Fig. 6. Reconstruction images from holograms produced by different methods. Highlighted insets are zoom-ins for detailed visualization. Numerical analysis is shown in Table 2.

6 EVALUATION

We validate our DPRC framework with several objective metrics for transmission and reconstruction quality. Specifically, we evaluate our framework's phase retrieval quality in Section 6.1, effectiveness in transmission data volume/latency reduction in Section 6.2, and edge-side compute/energy cost in Section 6.3. Furthermore, we also analyze the intra-system performance in Section 6.4 and several ablation studies in Section 6.5 to evaluate our system efficacy. We also validate the applicability of our framework and its performance on video sequences in Section 6.6. Finally, we demonstrate our method on experimental hardware prototype display and assess its performance in Section 6.7.

6.1 Phase Retrieval Quality

Objective Metrics. We use four metrics to evaluate the quality of reconstructed images from the retrieved phase holograms: peak signal-to-noise ratio (PSNR) and the recently proposed FLIP [Andersson et al. 2020] as difference evaluators, and the structural similarity index (SSIM) [Wang et al. 2004] and LPIPS [Zhang et al. 2018] as perceptual error metrics. Specifically, LPIPS measures the difference between features as computed by a pre-trained VGG [Simonyan and Zisserman 2015] network for any given two images, and FLIP similarly evaluates the perceptual difference by also considering the principles of human perception and incorporates dependencies on viewing distance and pixel size. A higher score is desired for PSNR and SSIM, whereas a lower is desired for LPIPS and FLIP. We evaluate our phase retrieval network (PRN) against the state-of-the-art non-iterative methods including Double Phase Amplitude Coding (DoublePhase) [Maimone et al. 2017] and HoloNet [Peng et al. 2020], and the iterative method WirtingerSGD [Chakravarthula et al. 2019; Peng et al. 2020] running for 500 iterations. The metrics are evaluated on 100 test images from the DIV2K dataset [Timofte et al. 2017] and the corresponding results are reported in Table 2.

Table 2. Reconstruction performance for phase retrieval

Method	PSNR ↑	SSIM ↑	LPIPS ↓	FLIP ↓	Time (s)
WirtingerSGD	**34.3434**	**0.9596**	**0.1299**	**0.0318**	247.88
DoublePhase	25.6460	0.7538	0.4291	0.1844	**0.013**
HoloNet	29.7014	0.9114	0.2394	0.1754	**0.027**
Ours (PRN)	**30.4155**	**0.9237**	**0.2006**	**0.1637**	**0.027**

Top two results are highlighted. Results for Time are calculated for generating 3-channel holograms.

Results. As shown in Table 2, WirtingerSGD shows the highest reconstruction quality on all four error metrics. However, iterative hologram computation takes more than 200s per frame. Among the other three real-time methods (less than 0.1s/frame), PRN shows the highest reconstruction quality among all the reported metrics. For instance, PRN is the only method achieving a PSNR > 30, with a runtime of less than 1/9000 of WirtingerSGD iterative optimization. DoublePhase on the other hand produces the lowest reconstruction quality, as evidenced by its low PSNR and SSIM values, *i.e.*, 25.65 and 0.7538 respectively. For LPIPS and FLIP metrics, the numerical results in Table 2 show a similar trend to PSNR and SSIM. Visual comparisons validating the above scores are provided in Figure 6. Additional examples and the corresponding difference visualizations produced by FLIP can be found in the Supplementary Material.

Dicussion. Although WirtingerSGD produces the highest reconstruction quality, it is prohibitively time inefficient in practice for real-time and interactive applications. On the other hand, our PRN phase retrieval network produces appealing results on all four metrics, as shown in Table 2, while demonstrating low running time. As for the reconstructed images, the holograms generated by PRN and WirtingerSGD produce apparently less artifacts than the only

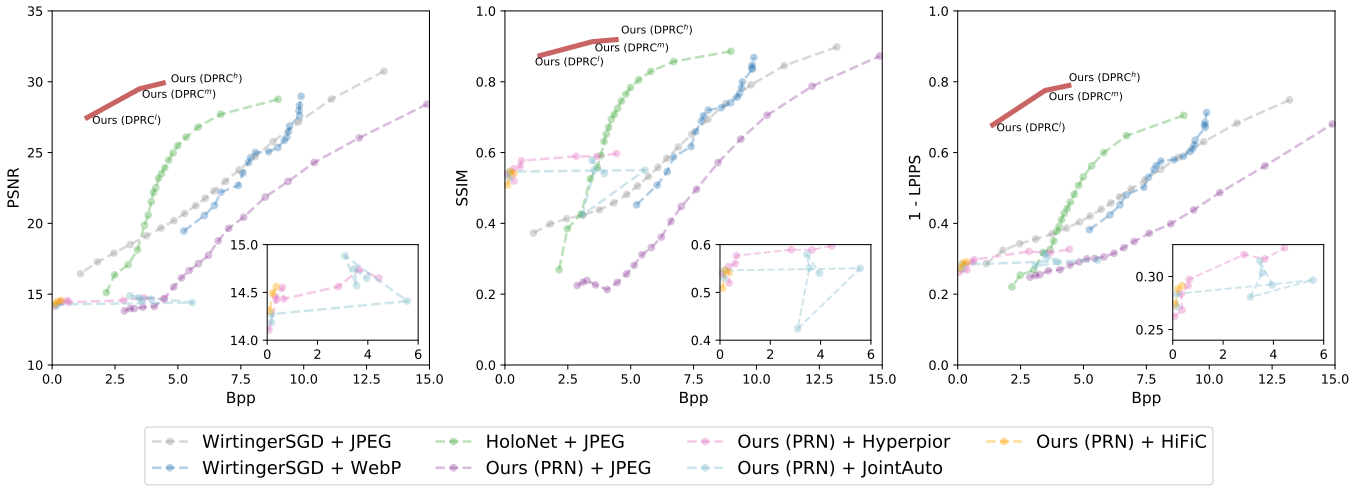

Fig. 7. Compression rate and quality curves of reconstruction results. Bpp represents bits per pixel used to encode the compressed holograms and + denotes compression. Note that results for (1-LPIPS) are given for consistency. X-/y-axis shows Bpps and the mean of quality values over 100 test images. **Larger is better for the values shown in y-axis and lower is better for Bpp.** The enlarged plot in each sub-figure is provided for better visualizing the performance for several alternatives.

other neural network-based approach HoloNet, especially in areas with flat textures as can be observed in Figure 6. These experiments validate that PRN achieves a better balance between quality optimization and time efficiency for holographic phase retrieval.

6.2 Transmission Efficiency

To make the proposed DPRC adapt to different Internet conditions, we trained our model with three quality levels by setting α_r to 1 (DPRC^l), 5 (DPRC^m), and 10 (DPRC^h), respectively, and analyze the results here.

Metrics and Conditions. Using bit-rates as the metric, we compare three DPRC-derived variants with two standard compression codecs including *JPEG* [Wallace 1992] and *WebP* [2], and three learning-based image compression methods including *Hyperprior* [Ballé et al. 2018], *JointAuto* [Minnen et al. 2018], and *HiFiC* [Mentzer et al. 2020].

Unlike our DPRC framework, holograms generated using other alternative conditions need to be compressed before transmission. To this end, we generate holograms using different methods discussed in Section 6.1, and thoroughly compare compression with *JPEG* and other neural network-based codecs. Specifically, we evaluate three conditions: *WirtingerSGD + JPEG*, *HoloNet + JPEG*, and *PRN + JPEG*. Figure 7 shows the rate-performance curves for the average values on 100 evaluation images. The compression performance is evaluated as the number of bits per pixel (bpp). Among the neural network-based compression frameworks, we use the pre-trained models provided by the authors of *HiFiC*, and the compressAI implementations[Bégaint et al. 2020] for *Hyperprior* and *JointAuto*, so that all of the implementations are implemented on PyTorch platform [Paszke et al. 2019]. Since [Ballé et al. 2018; Mentzer et al. 2020; Minnen et al. 2018] are trained on RGB images, the holograms for three channels are combined before applying the above methods.

[2]http://code.google.com/speed/webp/

Results. Figure 7 shows the statistical results of the above mentioned compression experiments. The DPRC condition shows significant performance gains over all other conditions (for example, >5 higher PSNR than all other conditions for the same Bpp levels). Note that DPRC always achieves lower than 5 bpp, hence the short red curves. DPRC^l achieves 27.42dB for PSNR and around 0.9 for SSIM with only 1.3 bpp (0.43 bpp per channel). In other words, DPRC demonstrates the reconstruction quality with an 18× compression ratio compared to the typical 24-bit Bitmap format. To achieve similar reconstruction quality, *HoloNet + JPEG* consumes around 7× more bits and *WirtingerSGD + JPEG* needs about 10× more bits.

Among the alternative conditions, *HoloNet + JPEG* compression shows the highest quality when bpp $\geq$ 5. As shown in Figure 7, it can also be observed that the *JPEG* codec significantly degrades the hologram reconstruction quality when bit-rates are lower than 7, especially for the LPIPS metric. Moreover, *WirtingerSGD + JPEG* provides significantly worse reconstruction quality than other compression alternatives at considerably lower bit-rates. Besides, it can be seen that the performance for *WirtingerSGD + WebP* is similar to that for *WirtingerSGD + JPEG* and shows only a small range of quality/bit-rate change. For holograms computed using our PRN network and compressed using learning-based methods (i.e. the three conditions including *PRN + HyperPrior*, *PRN + JointAuto* and *PRN + HiFiC*), the PSNR is lower than 15dB and SSIM is lower than 0.5, although the lowest bit-rates are achieved.

Sampled evaluation results are visualized in Figure 8. It can be seen that *JPEG* compression introduces noticeable artifacts in the reconstructed images whenever the bit-rates approach DPRC^h or higher. Additionally, Figure 9 provides example reconstructions from compressed holograms using the *HyperPrior*, *JointAuto* and *HiFiC* methods, and compare against our DPRC method. Specifically, in Figure 9, the results are produced with both the highest and the

Fig. 8. Reconstruction images from compressed holograms. DPRC^h achieves high quality reconstruction results with a similar bit-rate to JPEG (Qaulity:45), with which JPEG degrades holograms drastically.

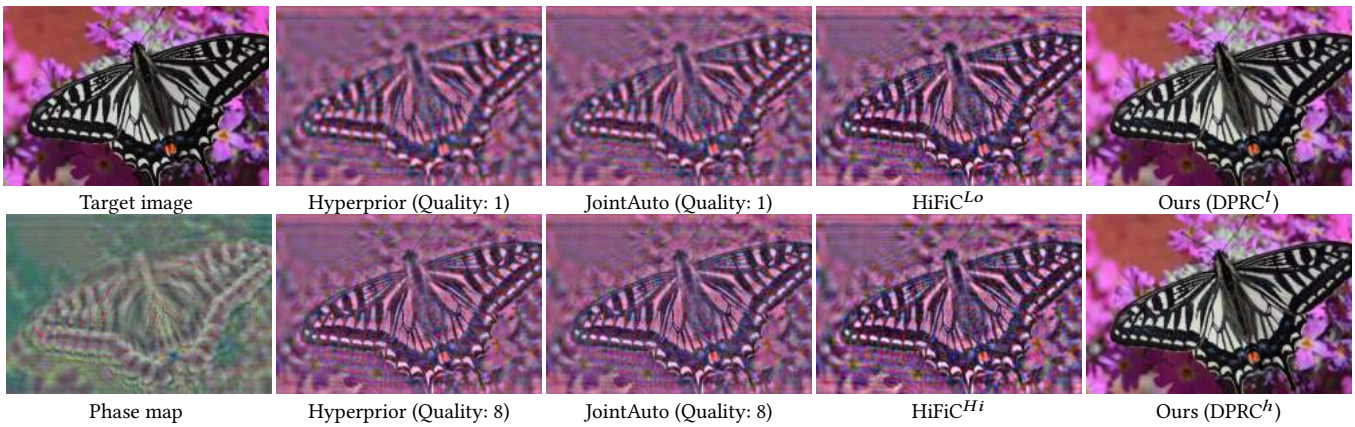

Fig. 9. Comparison with recent learning based image compression methods with two quality levels. For Hyperprior and JointAuto, the highest quality is 8 and the lowest quality is 1. For HiFiC, the highest and lowest quality are denoted as HiFiC^{Hi} and HiFiC^{Lo}.

lowest quality setting for each method. Despite the learning-based compression methods [Ballé et al. 2018; Mentzer et al. 2020; Minnen et al. 2018] being trained on image datasets with diverse content, it can be observed that the reconstruction quality is $\leq$ 15dB in PSNR when applied to hologram compression. In our ablation study given in Section 6.5, we will discuss about a learning based compression variant trained on pre-computed holograms.

Discussion. We observe that both statistical (Figure 7) and visual results (Figure 8) demonstrate that the proposed DPRC framework is effective in achieving high quality reconstruction using lower bit-rates, compared to existing alternatives. We speculate that the significant quality degradation with *WirtingerSGD + JPEG* condition is likely caused by the severe loss of high-frequency components on phase data, post compression. Note that while JPEG compression works well for low frequency natural image data, any information loss during compression of high frequency phase hologram data results in global noise and significant quality degradation of reconstructed images. A similar trend to *WirtingerSGD + JPEG* condition is also observed with *WirtingerSGD + WebP*. Figure 9 demonstrates that learning-based methods for natural image compression fail to directly generalize to hologram compression.

6.3 Computation and Energy Efficiency

Metrics and Conditions. Here, we breakdown our DPRC framework into modules and evaluate the model size, GPU compute cost, GPU energy consumption, and runtime of our framework. In particular, we compare the proposed DPRC framework with the other neural network alternative, HoloNet [Peng et al. 2020]. The model

size is measured as the number of parameters within the neural network, and compute cost is evaluated as GFLOPs (Giga floating-point operations) denoting the number of multiply-add operations. The GFLOPs of f_p^d (the propagation operator as described in Equation (2)) is excluded for both our DPRC and HoloNet due to the underlying non-transparent implementations, for example the FFT and IFFT operations as implemented in PyTorch. For energy consumption, we track GPU energy consumption during the network inference stage using the Carbontracker tool [Anthony et al. 2020]. The time taken for loading data from disks is not included for a fair comparison. All the above metrics are evaluated for generating single channel 1920×1080 phase hologram. During the experiment, we ran the networks for 1000 times and calculated the the average values.

Table 3. Computation cost and energy consumption on the edge side

Method	#Params.	GFLOPs	Energy(kJ)	Time (ms)
DPRC ($D_p + D_h$)	539, 393	154	21.72	3
HoloNet	2, 868, 754	656	131.56	9

Results. Table 3 reports the experimental results. The modules including the hyper-prior decoder D_h and the phase decoder D_p deployed on the edge side for DPRC have significantly lower number of parameters and lower computation cost than HoloNet in GFLOPs. Specifically, our framework resulted in a 17% reduction in energy consumption to reconstruct the holograms of an identical size on the edge device.

Discussion. The above experiment which ran on a single edge device demonstrates that DPRC shows significant compute- and energy-efficiency on edge devices without sacrificing performance. Under a real-world content streaming setting, the server may encode the hologram data only once, followed by distributed and concurrent edge-side decoding. Consequently, the overall savings in computation/energy can be further boosted when a large number of users and frames are simultaneously considered, as shown in Figure 10. In contrast to HoloNet which requires 2.5GB GPU memory during the inference stage, the decoders $D_p + D_h$ in DPRC consumes 1.5 GB memory, which lowers the requirements on client-side hardware.

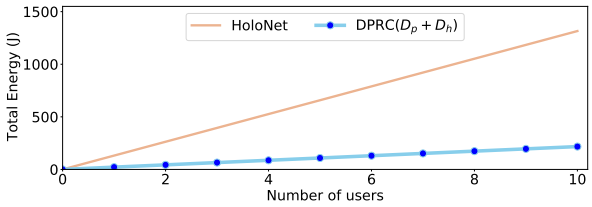

Fig. 10. Energy-User curves for HoloNet and DPRC.

6.4 Intra-System Analysis

In this section, we further analyze the system performance of the presented DPRC , including its decoding efficiency (Section 6.4.1) on the edge side and its robustness to potential contamination such as network package losses (Section 6.4.2).

6.4.1 Decoding Efficiency. We use inference time to measure the real-world decoding efficiency. The inference time includes time for decoding the data from bit-stream and data transmission cost between GPU and CPU. To eliminate the influence of I/O, we do not count the time for loading data from disks. The results are averaged over one hundred 1080×1920 images. The evaluation results are reported in Table 4.

Table 4. Decoding time for different compression methods

Method	JPEG ($Q = 100$)	Hyperprior	JointAuto	HiFiC	Ours
Decoding Time (s/channel)	0.0231	0.0719	16.1789	3.0077	0.0344

The time is averaged for 3 channels. $Q = 100$ denotes a quality level of 100.

As shown in Table 4, our method achieves a decoding rate of about 30 FPS for a single-channel phase hologram, which is much faster than *HiFiC*. This is because our DPRC network uses a single smaller network to predict the Gaussian mixture model, as opposed to HiFiC which adopts two larger hyper decoders and a slow implementation of arithmetic coding. DPRC method is faster than *Hyperprior*, which uses a similar entropy decoder as ours. This is because we adopt smaller number of channels for hyperlatent feature space $\mathbf{z}$. Moreover, we implemented a refined rANS coder [Duda 2014] with further reduced the data conversion cost. On the other hand, *JointAuto* is particularly slower compared to our method as it decodes pixels sequentially in an auto-regressive manner. Overall, although the performance of our method is slightly slower than that of *JPEG*, note that *JPEG* is a well-optimized standard codec with hardware acceleration and no data conversion cost between GPU and CPU. Therefore, our decoding process is still efficient in our prototype implementation, leaving much room for improvement.

6.4.2 Robustness Analysis. The contamination of transmission data over the cloud, such as Internet package loss, may affect the final image quality. This is particularly exacerbated in phase hologram transmission which is sensitive to information loss, as minor errors in phase data lead to noticeable global reconstruction errors. In this section, we investigate the robustness of our method to noise that might appear in the latent vector $\hat{\mathbf{v}}$ decoded from the bit-stream on the edge device. Specifically, we simulate the data contamination by adding noise (sampled from a normal distribution with zero-mean and different standard deviations σ_n) to the latent space $\hat{\mathbf{v}}$, and then evaluate the reconstruction quality of decoded holograms from the noisy latent space. The holographic reconstruction performance from the phase data decoded from the contaminated latent space is shown in Figure 11. Even though the phase decoder D_p receives the latent vector $\hat{\mathbf{v}}$ perturbed by random noise with different scales, the reconstruction performance is robust to noise sampled with σ_n in the range $[0.01, 0.5]$. This is possibly due to the fact that we train our DPRC with quantization, which can be approximated by adding uniform noise [Ballé et al. 2017]. To visualize this effect, we provide two examples in the supplementary material, demonstrating a consistent trend with the numerical results.

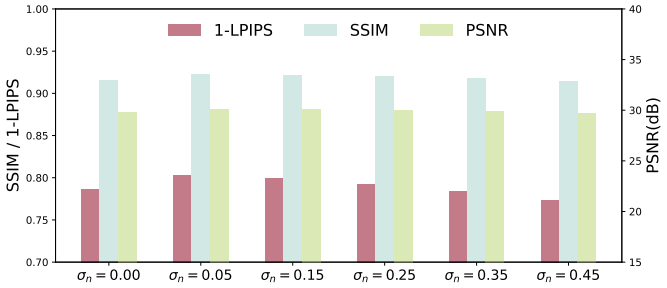

Fig. 11. Quantitative results evaluated on holograms produced using noisy latent representation, the higher the better.

Table 5. Performance for DPRC^h with varying decoder capacity

Variant	Bpp ↓	PSNR ↑	SSIM ↑	LPIPS ↓
DPRC^h($m = 1$)	4.3407	28.6116	0.8935	0.3061
DPRC^h($m = 2$)	**4.2573**	29.4576	0.9144	0.2241
DPRC^h($m = 3$)	4.2837	**29.6596**	**0.9153**	**0.2231**

6.4.3 Varying Decoder Capacity. We also study the effects of varying the network capacity of the phase decoder D_p, which is deployed on the edge device in DPRC framework. To this end, we apply the phase decoder D_p with varying network capacities to DPRC^h and report the reconstruction performance and bit-rates in Table 5. Note that DPRC^h adopts a phase decoder D_p with $m = 4$ residual blocks by default. As reported in Table 5, using $m = 3$ and $m = 2$ residual blocks lower the PSNR of reconstructed holograms only by 0.26dB and 0.46dB respectively. Therefore, the decoder D_p can be further shrunk in capacity for lower cost. Besides, when the number of residual blocks is reduced to $m = 1$, the degradation becomes apparent and is especially reflected on the perceptual LPIPS metric.

6.5 Ablation Studies

We conduct several ablation experiments to analyze the effectiveness of our DPRC framework in joint hologram generation and compression, the role of human visual system inspired perceptual loss Watson-DFT [Czolbe et al. 2020] in hologram generation, and the effect of the multi-scale structure for phase encoder E_p. All the experimental results are summarized in Table 6.

Effectiveness of DPRC Framework. As we are not aware of any previous method that has *coupled hologram generation and compression*, we investigate the effectiveness of the proposed DPRC framework by comparing it with two modified variants. 1) The first variant is to train a compression network taking pre-computed holograms as input and compress/decompress the holographic phase input. We supervise the network by the rate-generation loss calculated for recovered holograms and an additional hologram reconstruction loss during training. For training this variant, we prepared a hologram dataset that contains 3200 high-quality holograms calculated for (data augmented) 800 images of the DIV2K dataset using WirtingerSGD [Peng et al. 2020] method. 2) The second variant is to stack the state-of-the-art hologram generation network and a compression pipeline implemented with a latent phase encoder E_p, a hyper-prior encoder E_h, a hyper-prior decoder D_h and a phase decoder D_p. The second variant is trained in two stages, similar to DPRC.

Table 6. Results for ablation study

Method	Bpp ↓	PSNR ↑	SSIM ↑	LPIPS ↓	FLIP ↓
Ours (precomp. holograms)	7.3926	23.3426	0.6570	0.4753	0.2195
Ours (HoloNet+$\{E_{\{p,h\}}, D_{\{h,p\}}\}$)	**4.1385**	28.2606	0.8961	0.2462	0.1795
Ours (wo. watson-FFT)	4.3626	29.4534	0.9155	0.2235	0.1772
Ours (wo. multi-branch)	4.2220	29.1405	0.9086	0.2361	0.1771
Ours (DPRC^h)	4.4193	**29.9190**	**0.9190**	**0.2105**	**0.1707**

Each variant is trained with the same α_r as DPRC^h.

As can be seen in Table 6, our DPRC framework outperforms the other two modified variants. With regards to the first variant, as the phase holograms are optimized per image by the iterative methods, we speculate that modeling the data distribution within the compression module is challenging and results in reduced reconstruction performance, as shown in the first row of Table 6. The improved performance of the second variant as reported in the second row of Table 6 validates the effectiveness of joint training of hologram generation and compression networks. However, this variant stacks the hologram generation and compression networks, making the overall stacked-network heavier at runtime. Finally, our strategy of coupling both the generation and compression of holograms achieves the best performance and results in sufficiently low latency and high reconstruction quality.

Effectiveness of Visual System-based Perceptual Losses. We evaluate the effectiveness of human visual system-based (HVS) perceptual loss, especially the Watson-DFT loss [Czolbe et al. 2020], by removing $\mathcal{L}_{\text{wdft}}$ from the DPRC framework training, i.e. removing $\mathcal{L}_{\text{wdft}}$ from Equation (7) and Equation (13). The third row of Table 6 demonstrates that the performance degrades on each error metric when the Watson-DFT perceptual loss $\mathcal{L}_{\text{wdft}}$ is removed. This shows that considering an HVS based perceptual loss is effective in generating higher quality holograms.

Effectiveness of the Multi-scale Encoder. To investigate the influence of the multi-scale encoder employed in our framework, we design a variant that adopts a single-branch structure for the latent encoder E_p. Using a single-branch encoder degrades the performance of the recovered holograms, as reported in the fourth row of Table 6, validating the effectiveness of utilizing a multi-scale encoder.

6.6 Holographic Video Compression

In this section, we investigate the scalability of our framework to generation and compression of holograms for video frames. Specifically, a video dataset [Wang et al. 2017] is utilized to train and evaluate the redundancy-based holographic video compression as discussed in Section 4.3. The dataset [Wang et al. 2017] contains 220 5-second video clips with 1920 × 1080 resolution. We extract the frames from videos of the highest quality, among which 158

Fig. 12. Examples of reconstructed images from two holographic video compression solutions. The results produced from DPRC variants show less artifacts than HoloNet + H.264.

videos are used for training and other videos are used as validation set. Two versions of this variant are trained for jointly compressing every two frames or every three frames, which are indicated as Ours (DPRC , 2-joint) and Ours (DPRC , 3-joint) in Table 7. Finally, we evaluate the models on 15 videos and 18 consecutive frames are utilized from each video to avoid bias that might be introduced by videos with larger amount of frames. The results are reported in Table 7. As shown in Table 7, we also evaluate the performance

Table 7. Results for the pilot experiments on videos

Method	Bpp↓	PSNR↑	SSIM↑	LPIPS↓	FVVDP (JOD) ↑	Decoding Time (s/channel) ↓
HoloNet (H.264, Q=00)	8.9232	28.9409	0.8788	0.2927	6.8447	0.0729
HoloNet (H.264, Q=10)	1.6331	27.7853	0.8586	0.3622	6.6909	0.0287
Ours (DPRC^m)	2.4040	34.5842	0.9300	0.2094	8.5250	0.0275
Ours (DPRC, 2-Joint)	1.7387	33.5657	0.9325	0.2034	8.5009	0.0219
Ours (DPRC, 3-Joint)	1.3867	32.9540	0.9276	0.2372	8.3319	0.0185

The decoding time for H.264 is CPU time that has eliminated the time cost for I/O and waiting time among processes. Settings for computing the values on FVVDP metric: FovVideoVDP v1.0, 10.5 $[pix/deg]$, Lpeak=100, Lblack=0.4979$[cd/m^2]$, non-foveated.

of DPRC^m, which is trained for single images and has similar performance to models trained for joint compression on video frames. For evaluating the quality of reconstructed video frames, we also measure FVVDP [Mantiuk et al. 2021], which is a video difference metric that models multiple aspects of perception including spatial, temporal, and peripheral factors. As shown in Table 7, the average bit-rates and decoding time decrease with the increase in the number of consecutive frames compressed jointly, with reconstruction quality maintained. This demonstrates the potential of scaling DPRC to holographic video compression. In addition, we also compare against a naive alternative, *i.e*, compressing and decompressing the holograms produced from a state-of-the-art method HoloNet [Peng et al. 2020] via the prevalent video codec H.264. The compression is implemented via FFMPEG library[3] with the convention that a smaller quality number Q represents higher video quality, *e.g.* Q00 represents the highest video quality. As shown in Table 7, regardless of the bits per pixel, the performance of holographic video data generated by HoloNet and compressed by H.264 standard lags behind that of the recovered holograms from our DPRC method. The reconstructed video frames demonstrated in Figure 12 show more artifacts for decompressed holograms from HoloNet + H.264 compression than those from DPRC, validating the effectiveness our framework for holographic video compression.

[3]https://www.ffmpeg.org

6.7 Hardware-Captured Results

In addition to the evaluation on simulated results, we also conduct experiments on real hardware. Figure 13 shows examples of the real experimental results. We have tested on scenes with significant texture details, as can be seen, and the holograms decoded on the edge are able to produce the performance on par with that produced by other conventional methods, but requiring far less per pixel bit consumption. A few distortions such as ringing at the edges can be observed in the hardware captures making the reconstructions not on par with the simulations. We note that these arise due to the non-idealities in the hardware prototype and are not correlated to the framework itself. Note that the holograms produced by our DPRC framework use at least 3x lower bits per pixel compared to HoloNet or WirtingerSGD methods, but produces images that are comparable to state-of-the-art. While the WirtingerSGD optimization methods requires extensive compute, HoloNet suffers a loss of resolution that degrades the images noticeably as compared to our DPRC holograms. Note that incorporating recently proposed camera-in-the-loop strategies [Chakravarthula et al. 2020b; Peng et al. 2020] can further improve the captured image quality, but is beyond the scope of the current work.

7 CONCLUSIONS AND DISCUSSIONS

We presented the first end-to-end method for efficient generation and lossless transmission of phase holograms with high reconstruction quality. To this end, our method distributes the compute and transmission between the remote server and the edge client, similar to the existing cloud-based gaming services, to enable the future

Target image HoloNet WirtingerSGD Ours ($DPRC^h$)

Fig. 13. Results captured on a prototype holographic display. Note that our compressed holograms (right) only consumes 5 bits per pixel as opposed to HoloNet (middle left) and WirtingerSGD (middle right) optimized holograms which cost 14 and 22 bits per pixel, respectively. However, the performance of holograms produced by our DPRC method is comparable to the existing methods.

portable consumer-level holographic displays. We extensively evaluated our method and demonstrated its effectiveness on robust and accelerated holographic phase retrieval on the client end. In this work, we bring to attention the challenging problem of holographic phase compression and lossless retrieval, especially for future low-power everyday-use holographic displays on the edge client. We believe that our method motivates researchers to explore this new and exciting area.

Limitations and Future Work. The holographic video compression demonstrated in this paper only relies on redundancy. However, incorporating advanced video coding approaches would help achieve more significant quality/performance gains and is an exciting part of future work. Furthermore, we foresee extending the method to 3D assets; which typically incorporate higher dimensions, more complicated structures, and larger volumes; which would inspire interactive applications such as holographic gaming and teleportation. Recent advances in visualization use implicit neural representations to achieve impressive super-resolution performance [Chen et al. 2021] and efficient reduction of data volumes for 3D shapes [Davies et al. 2021; Martel et al. 2021]. Extending our framework in combination with implicit representation of the holographic content is another exciting future direction.

ACKNOWLEDGEMENTS

We would like to thank the anonymous reviewers for their constructive comments. We thank Shufeng Lin and Kexuan Liu for early discussions and validation. Praneeth Chakravarthula was supported by NSF grants 1840131 and 1405847. This work was supported in part by NSFC Projects of International Cooperation and Exchanges (62161146002); Shenzhen Collaborative Innovation Program (CJGJZD2021048092601003).

REFERENCES

Pontus Andersson, Jim Nilsson, Tomas Akenine-Möller, Magnus Oskarsson, Kalle Åström, and Mark D. Fairchild. 2020. FLIP: A Difference Evaluator for Alternating Images. In *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, Vol. 3. Article 15, 15:1–15:23 pages.

Lasse F. Wolff Anthony, Benjamin Kanding, and Raghavendra Selvan. 2020. Carbontracker: Tracking and Predicting the Carbon Footprint of Training Deep Learning Models. ICML Workshop on Challenges in Deploying and monitoring Machine Learning Systems.

Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. 2018. Variational image compression with a scale hyperprior. In *International Conference on Learning Representations (ICLR)*.

Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. 2016. End-to-end optimization of nonlinear transform codes for perceptual quality. In *Picture Coding Symposium (PCS)*. IEEE Signal Processing Society, 1–5.

Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. 2017. End-to-end optimized image compression. In *International Conference on Learning Representations (ICLR)*.

Jean Bégaint, Fabien Racapé, Simon Feltman, and Akshay Pushparaja. 2020. CompressAI: a PyTorch library and evaluation platform for end-to-end compression research. *arXiv preprint arXiv:2011.03029* (2020).

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. 2013. Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation. *arXiv preprint arXiv:1308.3432* (2013).

Stephen A Benton and V Michael Bove Jr. 2008. *Holographic imaging*. John Wiley & Sons.

Lokesh Boominathan, Mayug Maniparambil, Honey Gupta, Rahul Baburajan, and Kaushik Mitra. 2018. Phase retrieval for Fourier Ptychography under varying amount of measurements. *arXiv preprint arXiv:1805.03593* (2018).

Praneeth Chakravarthula, Yifan Peng, Joel Kollin, Henry Fuchs, and Felix Heide. 2019. Wirtinger Holography for Near-Eye Displays. *ACM Transactions on Graphics (TOG)* 38, 6, Article 213 (2019).

Praneeth Chakravarthula, Ethan Tseng, Henry Fuchs, and Felix Heide. 2022. Hogel-free Holography. *ACM Transactions on Graphics (TOG)* (2022).

Praneeth Chakravarthula, Ethan Tseng, Tarun Srivastava, Henry Fuchs, and Felix Heide. 2020a. Learned hardware-in-the-loop phase retrieval for holographic near-eye displays. *ACM Transactions on Graphics (TOG)* 39, 6 (2020), 1–18.

Praneeth Chakravarthula, Ethan Tseng, Tarun Srivastava, Henry Fuchs, and Felix Heide. 2020b. Learned Hardware-in-the-Loop Phase Retrieval for Holographic near-Eye Displays. *ACM Transactions on Graphics (TOG)* 39, 6, Article 186 (2020).

Praneeth Chakravarthula, Zhan Zhang, Okan Tursun, Piotr Didyk, Qi Sun, and Henry Fuchs. 2021. Gaze-Contingent Retinal Speckle Suppression for Perceptually-Matched Foveated Holographic Displays. *IEEE Transactions on Visualization and Computer Graphics* 27, 11 (2021), 41944203.

Rick H-Y Chen and Timothy D Wilkinson. 2009. Computer generated hologram from point cloud using graphics processor. *Applied optics* 48, 36 (2009), 6841–6850.

Yinbo Chen, Sifei Liu, and Xiaolong Wang. 2021. Learning Continuous Image Representation with Local Implicit Image Function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Mathew J Cherukara, Youssef SG Nashed, and Ross J Harder. 2018. Real-time coherent diffraction inversion using deep generative networks. *Scientific reports* 8, 1 (2018), 1–8.

Suyeon Choi, Manu Gopakumar, Yifan Peng, Jonghyun Kim, and Gordon Wetzstein. 2021. Neural 3D Holography: Learning Accurate Wave Propagation Models for 3D Holographic Virtual and Augmented Reality Displays. *ACM Trans. Graph. (SIGGRAPH Asia)* (2021).

Thomas M. Cover and Joy A. Thomas. 2006. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, USA.

Steffen Czolbe, Oswin Krause, Ingemar Cox, and Christian Igel. 2020. A Loss Function for Generative Neural Networks Based on Watson's Perceptual Model. *Advances in Neural Information Processing Systems* 33, 2051–2061.

Thomas Davies, Derek Nowrouzezahrai, and Alec Jacobson. 2021. On the Effectiveness of Weight-Encoded Neural Implicit 3D Shapes. *arXiv preprint arXiv:2009.09808* (2021).

Jarek Duda. 2014. Asymmetric numeral systems: entropy coding combining speed of Huffman coding with compression rate of arithmetic coding. *arXiv preprint arXiv:1311.2540* (2014).

M Hossein Eybposh, Nicholas W Caira, Mathew Atisa, Praneeth Chakravarthula, and Nicolas C Pégard. 2020. DeepCGH: 3D computer-generated holography using deep learning. *Optics Express* 28, 18 (2020), 26636–26650.

Alexandre Goy, Kwabena Arthur, Shuai Li, and George Barbastathis. 2018. Low photon count phase retrieval using deep learning. *Physical review letters* 121, 24 (2018).

Robert M Gray. 2011. *Entropy and information theory*. Springer Science & Business Media.

Yueyu Hu, Wenhan Yang, Zhan Ma, and Jiaying Liu. 2021. Learning end-to-end lossy image compression: A benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2021).

Shuming Jiao, Zhi Jin, Chenliang Chang, Changyuan Zhou, Wenbin Zou, and Xia Li. 2018. Compression of Phase-Only Holograms with JPEG Standard and Deep Learning. *Applied Sciences* 8, 8, Article 1258 (2018).

Michael R. Kellman, Emrah Bostan, Nicole A. Repina, and Laura Waller. 2019. Physics-Based Learned Design: Optimized Coded-Illumination for Quantitative Phase Imaging. *IEEE Transactions on Computational Imaging* 5, 3 (2019), 344–353.

Zachary David Cleary Kemp. 2018. Propagation based phase retrieval of simulated intensity measurements using artificial neural networks. *Journal of Optics* 20, 4 (2018), 045606.

Hwi Kim, Joonku Hahn, and Byoungho Lee. 2008. Mathematical modeling of triangle-mesh-modeled three-dimensional surface objects for digital holography. *Applied optics* 47, 19 (2008), D117–D127.

Seung-Cheol Kim and Eun-Soo Kim. 2008. Effective generation of digital holograms of three-dimensional objects using a novel look-up table method. *Appl. Opt.* 47, 19 (Jul 2008), D55–D62.

Xiangbo Li, Mahmoud Darwich, Magdy Bayoumi, and Mohsen Amini Salehi. 2020. Cloud-Based Video Streaming Services: A Survey. *arXiv preprint arXiv:2011.14976* (2020).

Robert LiKamWa, Zhen Wang, Aaron Carroll, Felix Xiaozhu Lin, and Lin Zhong. 2014. Draining Our Glass: An Energy and Heat Characterization of Google Glass. In *Proceedings of 5th Asia-Pacific Workshop on Systems*. ACM New York, NY, Article 10.

Siwei Ma, Xinfeng Zhang, Chuanmin Jia, Zhenghui Zhao, Shiqi Wang, and Shanshe Wang. 2020. Image and Video Compression With Neural Networks: A Review. *IEEE Transactions on Circuits and Systems for Video Technology* 30, 6 (2020), 1683–1698.

Andrew Maimone, Andreas Georgiou, and Joel S. Kollin. 2017. Holographic Near-Eye Displays for Virtual and Augmented Reality. *ACM Transactions on Graphics (TOG)* 36, 4, Article 85 (2017).

Rafał K. Mantiuk, Gyorgy Denes, Alexandre Chapiro, Anton Kaplanyan, Gizem Rufo, Romain Bachy, Trisha Lian, and Anjul Patney. 2021. FovVideoVDP: A Visible Difference Predictor for Wide Field-of-View Video. *ACM Transactions on Graphics (TOG)* 40, 4, Article 49 (2021).

Julien N.P. Martel, David B. Lindell, Connor Z. Lin, Eric R. Chan, Marco Monteiro, and Gordon Wetzstein. 2021. ACORN: Adaptive Coordinate Networks for Neural Representation. *ACM Transactions on Graphics (TOG)* 40, 4, Article 58 (2021).

Nobuyuki Masuda, Tomoyoshi Ito, Takashi Tanaka, Atsushi Shiraki, and Takashige Sugie. 2006. Computer generated holography using a graphics processing unit. *Optics Express* 14, 2 (2006), 603–608.

Kyoji Matsushima. 2005. Computer-generated holograms for three-dimensional surface objects with shade and texture. *Applied optics* 44, 22 (2005), 4607–4614.

Kyoji Matsushima and Tomoyoshi Shimobaba. 2009. Band-Limited Angular Spectrum Method for Numerical Simulation of Free-Space Propagation in Far and Near Fields. *Optics express* 17, 22 (2009), 19662–19673.

Fabian Mentzer, George Toderici, Michael Tschannen, and Eirikur Agustsson. 2020. High-Fidelity Generative Image Compression. In *Advances in Neural Information Processing Systems*, Vol. 33. 11913–11924.

David Minnen, Johannes Ballé, and George Toderici. 2018. Joint Autoregressive and Hierarchical Priors for Learned Image Compression. In *Advances in neural information processing systems*. 1079410803.

Y. Ogihara and Y. Sakamoto. 2015. Fast calculation method of a CGH for a patch model using a point-based method. *Applied Optics* 54, 1 (2015), A76–A83.

Nitish Padmanaban, Yifan Peng, and Gordon Wetzstein. 2019. Holographic near-eye displays based on overlap-add stereograms. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–13.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*. 8024–8035.

Yifan Peng, Suyeon Choi, Nitish Padmanaban, and Gordon Wetzstein. 2020. Neural Holography with Camera-in-the-Loop Training. *ACM Transactions on Graphics (TOG)* 39, 6, Article 185 (2020).

Christoph Petz and Marcus Magnor. 2003. Fast hologram synthesis for 3D geometry models using graphics hardware. In *Proc. SPIE 5005, Practical Holography XVII and Holographic Materials IX*. 266–275.

Jorma Rissanen and Glen Langdon. 1981. Universal modeling and coding. *IEEE Transactions on Information Theory* 27, 1 (1981), 12–23.

Yair Rivenson, Yibo Zhang, Harun Günaydın, Da Teng, and Aydogan Ozcan. 2018. Phase recovery and holographic image reconstruction using deep learning in neural networks. *Light: Science & Applications* 7, 2 (2018), 17141.

Liang Shi, Beichen Li, Changil Kim, Petr Kellnhofer, and Wojciech Matusik. 2021. Towards real-time photorealistic 3D holography with deep neural networks. *Nature* 591, 7849 (2021), 234–239.

Tomoyoshi Shimobaba, Nobuyuki Masuda, and Tomoyoshi Ito. 2009. Simple and fast calculation algorithm for computer-generated hologram with wavefront recording plane. *Optics letters* 34, 20 (2009), 3133–3135.

K. Simonyan and A. Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conference on Learning Representations (ICLR)*.

David Taubman and Michael Marcellin. 2013. *JPEG2000 Image Compression Fundamentals, Standards and Practice*. Springer Publishing Company, Incorporated.

Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. 2017. Lossy Image Compression with Compressive Autoencoders. In *International Conference on Learning Representations (ICLR)*.

Radu Timofte, Eirikur Agustsson, Luc Van Gool, Ming-Hsuan Yang, Lei Zhang, Bee Lim, et al. 2017. NTIRE 2017 Challenge on Single Image Super-Resolution: Methods and Results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Gregory K Wallace. 1992. The JPEG still picture compression standard. *IEEE Transactions on Consumer Electronics* 38, 1 (1992), xviii–xxxiv.

Haiqiang Wang, Ioannis Katsavounidis, Jiantong Zhou, Jeonghoon Park, Shawmin Lei, Xin Zhou, Man-On Pun, Xin Jin, Ronggang Wang, Xu Wang, Yun Zhang, Jiwu Huang, Sam Kwong, and Kuo C.-C. Jay. 2017. VideoSet: A large-scale compressed video quality dataset based on JND measurement. *Journal of Visual Communication and Image Representation* 46 (2017), 292–302.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing* 13, 4 (2004), 600–612.

Z. Wang, E. P. Simoncelli, and A. C. Bovik. 2003. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems Computers*, Vol. 2. 1398–1402.

Hao Zhang, Liangcai Cao, and Guofan Jin. 2017. Computer-generated hologram with occlusion effect using layer-based processing. *Appl. Opt.* 56, 13 (May 2017), F138–F143.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 586–595.