Supplemental Document

Deep-learning-based fringe-pattern analysis with uncertainty estimation: supplement

SHIJIE FENG,[1,2,3] CHAO ZUO,[1,2,*] YAN HU,[1,2] YIXUAN LI,[1,2] AND QIAN CHEN[2,4]

[1] *Smart Computational Imaging (SCI) Laboratory, Nanjing University of Science and Technology, Nanjing, Jiangsu Province 210094, China*
[2] *Jiangsu Key Laboratory of Spectral Imaging & Intelligent Sense, Nanjing University of Science and Technology, Nanjing, Jiangsu Province 210094, China*
[3] *e-mail: shijiefeng@njust.edu.cn*
[4] *e-mail: chenqian@njust.edu.cn*
[*] *Corresponding author: zuochao@njust.edu.cn*

Supplement DOI: https://doi.org/10.6084/m9.figshare.16918204

Parent Article DOI: https://doi.org/10.1364/OPTICA.434311

Deep learning based fringe-pattern analysis with uncertainty estimation: supplemental document

This document provides the information about the theory and the implementation of the Bayesian convolutional neural network (BNN) using Concrete dropout, the optical set-up, the prediction of the phase and the uncertainty maps, the evaluation of predicted uncertainty, and other supplementary contents to the primary manuscript "Deep learning based fringe-pattern analysis with uncertainty estimation".

1. THE BNN APPROXIMATED BY CONCRETE DROPOUT AND DETAILS ABOUT ITS PREDICTION

To calculate the uncertainty of a neural network, people usually place a priori probability distribution on the weights of the neural network. BNN replaces the deterministic network's weight parameters with probability distributions over them. Here, we present a BNN that uses the Concrete dropout [1] to approximate Bayesian inference in deep Gaussian processes for learning the numerator $M(x,y)$ and the denominator $D(x,y)$ statistically. We assume that $\mathbf{X}$ is a set of input fringe images, which can be represented as $\mathbf{X} = \left\{\mathbf{x}^k\right\}_{k=1}^{K}$ where $\mathbf{x}^k$ is the kth input fringe pattern and K the size of the set. $\mathbf{Y}$ is a set of ground-truth labels corresponding to the training data, which can be written as $\mathbf{Y} = \left\{\mathbf{y}^k\right\}_{k=1}^{K}$ where $\mathbf{y}^k$ consists of the ground-truth numerator and denominator (M^k, D^k). $\mathbf{w}$ represents the weight matrix of the BNN. To investigate the distribution of the output of BNN, we model the predictive distribution $p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y})$ as

$$p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y}) = \int p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{w}) p(\mathbf{w}|\mathbf{X}, \mathbf{Y})\, d\mathbf{w}, \tag{S1}$$

where $p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{w})$ is the probability of the output $\mathbf{y}^*$ given the input $\mathbf{x}^*$, the weights $\mathbf{w}$ and $p(\mathbf{w}|\mathbf{X}, \mathbf{Y})$ the probability of the weights $\mathbf{w}$ given the training data $(\mathbf{X}, \mathbf{Y})$. The distribution $p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{w})$ describes the data uncertainty and the distribution $p(\mathbf{w}|\mathbf{X}, \mathbf{Y})$ characterizes the model uncertainty.

To measure the data uncertainty, we assume that $\mathbf{y}^k$ has N pixels, $p\left(\mathbf{y}^k|\mathbf{x}^k, \mathbf{w}\right)$ can then be written as

$$p\left(\mathbf{y}^k|\mathbf{x}^k, \mathbf{w}\right) = \prod_{i=1}^{N} p\left(\mathbf{y}_i^k|\mathbf{x}^k, \mathbf{w}\right). \tag{S2}$$

Assume that the distribution of $\mathbf{y}^k$ is Gaussian for pixel i, $p\left(\mathbf{y}_i^k|\mathbf{x}^k, \mathbf{w}\right)$ can be calculated as

$$p\left(\mathbf{y}_i^k|\mathbf{x}^k, w\right) = \frac{1}{\sqrt{2\pi}\sigma_i^k} \exp\left[-\frac{\left(\mathbf{y}_i^k - \mu_i^k\right)^2}{2\left(\sigma_i^k\right)^2}\right], \tag{S3}$$

where the pixels are assumed to be independent, and μ_i^k is the mean and σ_i^k the standard deviation of the pixel. The data uncertainty can be captured by minimizing the negative log-likelihood function at the training stage

$$-\frac{1}{K}\sum_k \log p\left(\mathbf{y}^k|\mathbf{x}^k, \mathbf{w}\right) = \frac{1}{K}\sum_{k=1}^{K}\left[\frac{1}{2\left(\sigma^k\right)^2}\left\|\mathbf{y}^k - \hat{\mathbf{y}}^k\right\|^2 + \frac{1}{2}\log\left(\sigma^k\right)^2\right], \tag{S4}$$

where $\mathbf{y}$ is the ground truth label, $\hat{\mathbf{y}}$ the result predicted by BNN, and σ^2 the predicted variance. Our BNN learns to output $\hat{\mathbf{y}}$ and σ^2 during the training. It is noteworthy that it is not necessary to collect the ground-truth variance σ^2. It can be learned automatically by the minimizing the Eq. (S4).

To measure the model uncertainty, the Concrete dropout network is applied [1]. Here, the dropout rate of each layer is no longer fixed and can be tuned automatically in the training stage.

By placing the Concrete dropout before every weight layer, we can use a simple variational distribution $q(\mathbf{w})$ to approximate $p(\mathbf{w}|\mathbf{X},\mathbf{Y})$ that is usually hard to calculate analytically. By using the Monte Carlo (MC) integration over T samples satisfying $\mathbf{w}^{(t)} \sim q(\mathbf{w})$, Eq. (S1) can be approximated as

$$p(\mathbf{y}^*|\mathbf{x}^*,\mathbf{X},\mathbf{Y}) \approx \int p(\mathbf{y}^*|\mathbf{x}^*,\mathbf{w}) q(\mathbf{w})\, d\mathbf{w} \approx \frac{1}{T}\sum_{t=1}^{T} p\left(\mathbf{y}^*|\mathbf{x}^*,\mathbf{w}^{(t)}\right). \tag{S5}$$

At the training stage, the loss function of our BNN can be defined according to the variational interpretation as [1]

$$L_{BNN} = -\frac{1}{K}\sum_{k=1}^{K} \log p\left(\mathbf{y}^k|\mathbf{x}^k,\mathbf{w}\right) + \frac{1}{K} KL\left[q(\mathbf{w})\,||p(\mathbf{w}|\mathbf{X},\mathbf{Y})\right]. \tag{S6}$$

where the first term is the mentioned negative log-likelihood function and the second term is the Kullback–Leibler(KL) divergence that be used to minimize the difference between the selected simple distribution and the actual distribution. The drop rate is learned with convergence of the KL divergence. Regarding the second term of L_{BNN}, since we adopt a simple distribution $q(\mathbf{w})$ to approximate the complex $p(\mathbf{w}|X,Y)$, we hope that the difference between them could be as smaller as possible. The distribution $q(\mathbf{w})$ can be written as

$$q(\mathbf{w}) = \prod_l q_{w_l^d}(w_l). \tag{S7}$$

And $q_{w_l^d}(w_l)$ can be expressed as

$$q_{w_l^d}(w_l) = w_l^d \cdot diag\left[Bernoulli(1-p_l)^{S_l}\right], \tag{S8}$$

where L is the number of dropout layers, w_l^d the deterministic weights of the lth layer, w_l the stochastic weights obtained after applying the dropout on w_l^d. p_l is the dropout rate of the lth layer and S_l the size of weights w_l. The KL divergence in Eq. (S6) can be approximate according to [1]

$$KL\left[q(\mathbf{w})\,||p(\mathbf{w}|\mathbf{X},\mathbf{Y})\right] = \sum_{l=1}^{L} KL\left[q_{w_l^d}(w_l)\,||p(w_l)\right], \tag{S9}$$

$$KL\left[q_{w_l^d}(w_l)\,||p(w_l)\right] = \lambda_1 \frac{l^2(1-p_l)}{2}\left\|w_l^d\right\|^2 - \lambda_2 S_l H(p_l), \tag{S10}$$

where λ_1 is the regularizer for the weights and λ_2 the regularizer for the dropout. $H(p_l)$ is the entropy of a Bernoulli random variable with probability p_l that can be written as

$$H(p_l) = -p_l \log p_l - (1-p_l)\log(1-p_l). \tag{S11}$$

In Concrete dropout, the Concrete distribution is a continuous distribution that is applied to approximating discrete Bernoulli random variables. For traditional dropout where the discrete Bernoulli distribution is applied, the random variable can only take 0 or 1. In this work, however, we sample realizations from the Concrete distribution with some temperature α which results in fractional values between 0 and 1. For the discrete Bernoulli random variable z, the Concrete distribution relaxation z_c can be parameterized as a sigmoid function

$$z_c = sigmoid\left\{\frac{1}{\alpha}\left[\log p_l - \log(1-p_l) + \log u - \log(1-u)\right]\right\}, \tag{S12}$$

where the random variable u has an uniform distribution, i.e., $u \sim U(0,1)$. The temperature α is positive and a small value is suggested for our applications (here, the α is selected as $2/3$ empirically). By the Concrete dropout, the dropout probability of each layer can be optimized during the training of BNN. Further experimental results corresponding to the learned dropout rates are provided in the section of Dropout rates learned by BNN.

At the prediction stage, the diagram is shown in the Fig. 1 of the main manuscript. The dropout layers in our BNN randomly set input neurons to zero with learned dropout rates. By collecting

the results of stochastic forward propagation through the trained model, the predictive mean can be computed and be used as the prediction of BNN

$$\hat{\mu} = E\left(\mathbf{y}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y}\right) \approx \frac{1}{T}\sum_{t=1}^{T} E\left(\mathbf{y}^*|\mathbf{x}^*, \mathbf{w}^{(t)}\right) = \frac{1}{T}\sum_{t=1}^{T} \mathbf{y}^{*(t)}, \tag{S13}$$

where E is the expectation. The model uncertainty is measured by the variance of the predicted results:

$$\hat{\sigma}^{model} = \sqrt{E\left[Var\left(\mathbf{y}^*|\mathbf{w}, \mathbf{x}^*, \mathbf{X}, \mathbf{Y}\right)\right]} \approx \sqrt{\frac{1}{T}\sum_{t=1}^{T}\left(\mathbf{y}^{*(t)} - \hat{\mu}\right)^2}. \tag{S14}$$

Then, the data uncertainty is quantified by the average of the estimated variance:

$$\hat{\sigma}^{data} = \sqrt{Var\left(E\left[\mathbf{y}^*|\mathbf{w}, \mathbf{x}^*, \mathbf{X}, \mathbf{Y}\right]\right)} \approx \sqrt{\frac{1}{T}\sum_{t=1}^{T}\left(\sigma^2\right)^{(t)}}. \tag{S15}$$

According to Eq. (S13), we can calculate the $\hat{\mu}_M$ and $\hat{\mu}_D$ over the T pairs of M and D by using

$$\hat{\mu}_M = \frac{1}{T}\sum_{t=1}^{T} M^{*(t)}, \tag{S16}$$

$$\hat{\mu}_D = \frac{1}{T}\sum_{t=1}^{T} D^{*(t)}, \tag{S17}$$

where $M^{*(t)}$ and $D^{*(t)}$ is the numerator and the denominator predicted by the model at tth MC sampling respectively. By substituting $\hat{\mu}_M$ and $\hat{\mu}_D$ into the arctangent function, we obtain the final wrapped phase $\hat{\mu}_\varphi$ by

$$\hat{\mu}_\varphi = \arctan\frac{\hat{\mu}_M}{\hat{\mu}_D}. \tag{S18}$$

Then, according to Eqs. (S14) and (S15) the numerator's model uncertainty and data uncertainty can be calculated by

$$\hat{\sigma}_M^{model} = \sqrt{\frac{1}{T}\sum_{t=1}^{T}\left(M^{*(t)} - \hat{\mu}_M\right)^2}. \tag{S19}$$

$$\hat{\sigma}_M^{data} = \sqrt{\frac{1}{T}\sum_{t=1}^{T}\left(\sigma_M^2\right)^{(t)}}, \tag{S20}$$

The uncertainties of the denominator can be obtained in the same way. Next, with the model/data uncertainty of the numerator and denominator, we can compute the corresponding uncertainties of the phase using the propagation of uncertainty

$$\hat{\sigma}_\varphi^{model} = \sqrt{\left(\frac{\partial\varphi}{\partial M}\hat{\sigma}_M^{model}\right)^2 + \left(\frac{\partial\varphi}{\partial D}\hat{\sigma}_D^{model}\right)^2}, \tag{S21}$$

$$\hat{\sigma}_\varphi^{data} = \sqrt{\left(\frac{\partial\varphi}{\partial M}\hat{\sigma}_M^{data}\right)^2 + \left(\frac{\partial\varphi}{\partial D}\hat{\sigma}_D^{data}\right)^2}, \tag{S22}$$

where $\hat{\sigma}_\varphi^{data}$ is the data uncertainty of the phase and $\hat{\sigma}_\varphi^{model}$ the model uncertainty of the phase, and $\varphi = \arctan\left(M/D\right)$.

Actually, one may also calculate the T wrapped phase maps with T pairs of M and D and then take the average over them for the mean wrapped phase. We show the error of this strategy with our experiment where the condition is the same as the one of the main manuscript. The test objects are two plaster models. The absolute phase error is shown in Fig. S1 (a). We can see that there are many points that are calculated with obvious errors as can be found from our selected ROIs. On contrary, these errors can be avoided with the proposed method as shown in Fig. S1 (b).

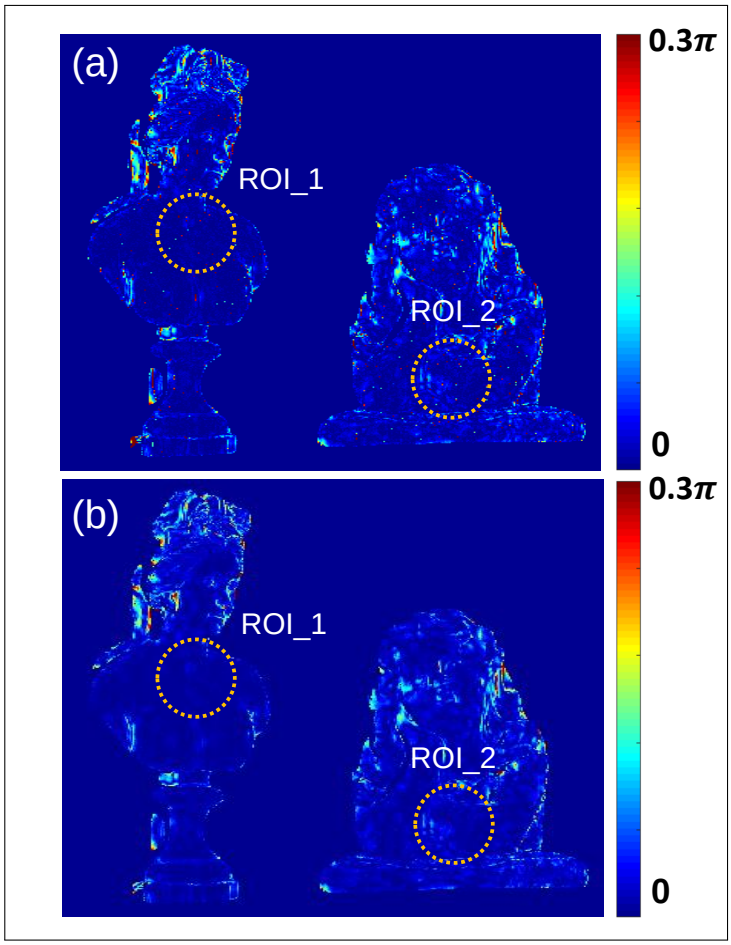

Fig. S1. Comparison of the absolute phase error. (a) The phase error of the unwrapped phase that is obtained by unwrapping the mean of T wrapped phase maps calculated by T pairs of M and D. (b) The phase error of the proposed method.

2. THE STRUCTURE AND THE IMPLEMENTATION OF BNN

Our BNN is constructed based on the architecture of the U-Net that is a fully convolutional networks with a encoder and a decoder [2]. Figure S2 shows the architecture of the BNN, in which we add Concrete dropout layers prior to every convolutional layers. The input of the network is a fringe image and we train the model to learn to predict the numerator M, the denominator D and their variance maps. To be specific, the input fringe image in our experiments has resolution of $W \times H$. It is then handled by the encoder, where C channels of features are extracted from the fringe pattern (C=50 in our experiments) by the convolutional layer that is activated by the rectified linear unit (ReLU) as activation function, i.e., $\mathrm{ReLU}(x) = \max(0, x)$. The extracted features are then down-sampled by $1/2$ along both x and y directions. With the next several similar down-sampling convolutional blocks that extract more image details at different scales, high-level features of $\frac{W}{16} \times \frac{H}{16} \times 16C$ are obtained by the encoder at last. They are then handled by the decoder that performs up-sampling to synthetize and recover the final results of the input image's original size. The up-sampling is carried out by bilinear interpolation that is followed by some Concrete convolutional layers. A skip connection can be found at each step of the decoder, which is used to concatenate the convolutional layers' output with feature maps from the encoder at the same level. Features at different levels can be collected at the same time with this architecture. The last layer of our BNN is a merged layer that include two convolutional layers. One is linearly activated and learns to output the numerator and the denominator. The other one is activated by the softplus function (i.e., $softplus\,(x) = log\,[exp\,(x) + 1]$) and learns to predict the variance of the numerator and denominator. The loss function is shown by Eq. (S6). It is noteworthy that we only need to provide the ground-truth data for the numerator and the denominator and the variance can be learned automatically.

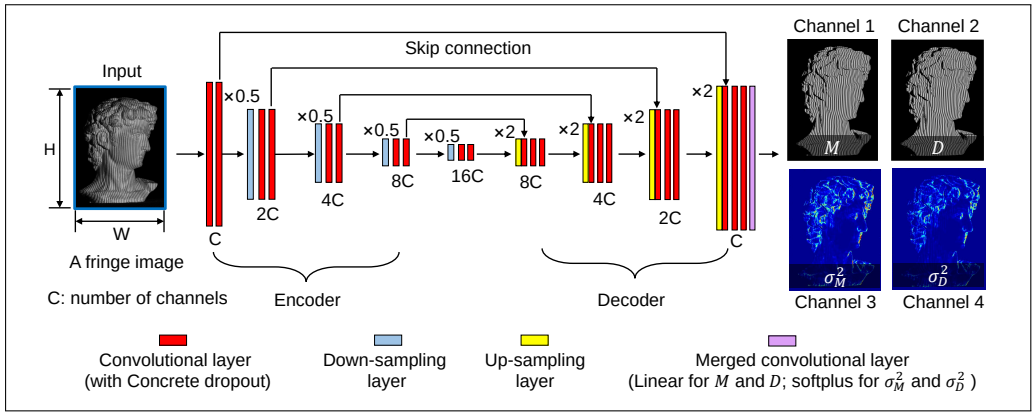

Fig. S2. The architecture of the proposed BNN that is built by modifying the U-Net. The input is a single fringe pattern. The model learns to predict a four-channel tensor that includes the numerator (M), the denominator (D) and their variance maps. Different from the architecture of the traditional U-Net, the BNN has concrete dropout layers prior to every convolutional layers. The dropout rates are learned during the training stage and are activated in the testing stage.

To train the BNN, we captured fringe images of different kinds of objects. Our training dataset consisted of 984 different fringe patterns, in which about 80% images were used for training and the rest for validation. The ground-truth labels for each fringe pattern were computed with the 12-step PS algorithm. Before being fed into the BNN, the input fringe pattern was divided by 255 for normalization, which can make the learning easier for the network. In the training stage of BNN, the adaptive moment estimation (ADAM) was used to tune the parameters to find the minimum of the loss function [3]. In the implementation of ADAM, we started the training with a learning rate of 10^{-4}. We dropped it by a factor of 2 if the validation loss has stopped improving for 10 epochs, which helps the loss function get out of local minima during training. All the data processing and the DNN's training and testing were implemented in Python by using Keras. The number of MC sampling was 50. The dropout rate of each layer (22 layers in our BNN) was learned by BNN during the training. Empirically, $\lambda_1 = 10^{-6}$ and $\lambda_2 = 10^{-5}$ were used in this work. In each layer of BNN, the dropout rate was initialized randomly according

to a uniform distribution $[0.2, 0.6]$ and then it was updated within the range $[0.0, 1.0]$ during the training process. We further demonstrate experimental results of the learned dropout rates in the section of Dropout rates learned by BNN.

3. OPTICAL SET-UP AND THE CALCULATION OF GROUND-TRUTH DATA

The fringe projection system consists of a projector (DLP 4100, Texas Instruments) with resolution of 1024×768 and a camera (V611, Vision Research Phantom) with resolution of 1280×800. The projector illuminates test objects with pre-designed fringe patterns and the camera captures the images simultaneously from a different perspective. The captured fringe patterns are 8-bit grayscale images. The schematic is shown in Fig. S3.

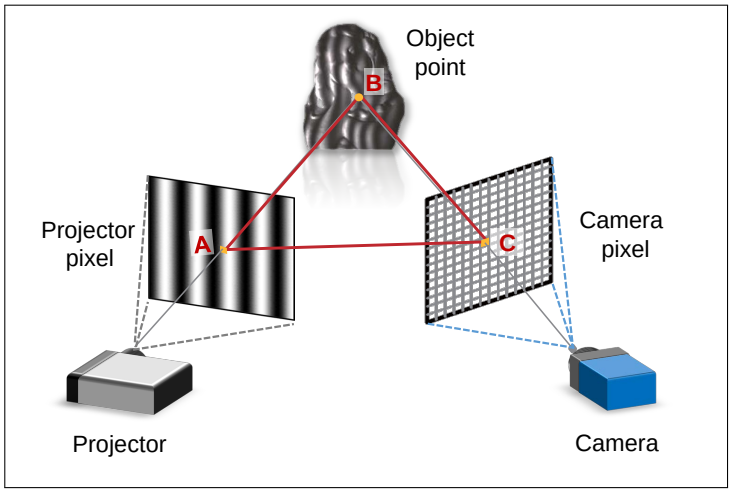

Fig. S3. Schematic of the fringe projection system.

To generate ground-truth labels, the 12-step phase-shifting (PS) algorithm was applied [4]. The projected 12-step PS fringe patterns can be written as

$$I_n^p(x^p, y^p) = a + b\cos\left(2\pi f x^p - \frac{2\pi n}{N_{ps}}\right), \tag{S23}$$

where (x^p, y^p) is the pixel coordinate of the projector, $N_{ps} = 12$, and $n = 0, 1, 2, ..., 11$. Parameters a, b, f are the DC component, amplitude and spatial frequency of the fringe pattern, respectively. The spatial frequency of the projected fringes is $f = 160$ in our experiments. These PS patterns are projected onto a test object and the captured images can be expressed as

$$I_n(x, y) = A(x, y) + B(x, y)\left[\varphi(x, y) - \frac{2\pi n}{N_{ps}}\right], \tag{S24}$$

where (x, y) is the pixel coordinate of the camera, A the background signal, B the modulation, and φ the phase of test objects. The ground-truth numerator and denominator (in Eq. (2) of the primary manuscript) can be calculated as

$$M(x, y) = \sum_{n=0}^{N_{ps}-1} I_n(x, y)\sin\left(\frac{2\pi n}{N_{ps}}\right), \tag{S25}$$

$$D(x, y) = \sum_{n=0}^{N_{ps}-1} I_n(x, y)\cos\left(\frac{2\pi n}{N_{ps}}\right). \tag{S26}$$

4. EVALUATION OF PREDICTED UNCERTAINTY

The numerator and the denominator are predicted by BNN directly and the phase is obtained by substituting them into the arctangent function. Since the uncertainties of phase are calculated with the propagation of uncertainty, it is not easy to evaluate the phase uncertainties directly.

Therefore, we tend to evaluate the accuracy of the predicted uncertainty of the numerator and the denominator which are assume to have Gaussian distribution in our method. The phase uncertainty can then be considered reliable if the uncertainty of the numerator and the denominator are correct. To evaluate the uncertainty of the numerator and the denominator predicted by BNN, the reliability diagram is computed [5], which demonstrates how the empirical probability of the ground truth matches the predicted results. The probability density of a pixel i to take an estimated value y is

$$f_i\left(\mathbf{y}^* = y\right) = p\left(\mathbf{y}^* = y \,|\mathbf{x}^*, \mathbf{X}, \mathbf{Y}\right) \approx \frac{1}{T}\sum_{t=1}^{T} G\left(y; \mu_i^t, \sigma_i^t\right). \tag{S27}$$

By defining the credible interval $C_i^\varepsilon = \left[u_i^t - \varepsilon, u_i^t + \varepsilon\right]$. The credibility p_i^ε is the estimated probability that the true mean falls within C_i^ε

$$p_i^\varepsilon = \int_{\mu_i^t-\varepsilon}^{\mu_i^t+\varepsilon} f_i\left(y\right) = \frac{1}{T}\sum_{t=1}^{T}\left[F\left(\mu_i^t + \varepsilon\right) - F\left(\mu_i^t - \varepsilon\right)\right] \tag{S28}$$

where F is the cumulative distribution function of the tth predicted Gaussian distribution from the Monte Carlo sampling. The bound ε can be determined by the error propagation of the image noise.

To create the reliability diagram, we define the bin interval Δp and the number of bins is $1/\Delta p$. The mth bin P_m falls within $(p_{m-1}, p_m]$. The averaged credibility $Cred\left(P_m, \varepsilon\right)$ takes the average over the set of pixels S_m^ε that have similar credibility within $(p_{m-1}, p_m]$

$$Cred\left(P_m, \varepsilon\right) = \frac{1}{N_{S_m^\varepsilon}} \sum_{i \in (p_{m-1}, p_m]} p_i \tag{S29}$$

where $N_{S_m^\varepsilon}$ is the number of pixels in the set S_m^ε. The empirical accuracy $Acc\left(P_m, \varepsilon\right)$ can be defined as the fraction of the pixels in the set S_m^ε where the true mean μ_i^* is within the corresponding credible intervals C_i^ε

$$Acc\left(P_m, \varepsilon\right) = \frac{1}{N_{S_m^\varepsilon}} \sum_{i \in (p_{m-1}, p_m]} \mathbf{I}_{\left\{\hat{\mu}_i - \varepsilon \le \mu_i^* \le \hat{\mu}_i + \varepsilon\right\}} \tag{S30}$$

In theory, the true mean is unknown and is approximated by the results obtained by 12-step phase shifting algorithm in our experiments.

To analyze the predicted uncertainty for the numerator and the denominator, we created the reliability diagrams for them respectively. The bound of the numerator ε_M and the denominator ε_D can be obtained by

$$\varepsilon_M = \sqrt{\sum_{n=1}^{N_{ps}} \left(\frac{\partial M}{\partial I_n}\right)^2 \sigma_{I_n}^2} \tag{S31}$$

$$\varepsilon_D = \sqrt{\sum_{n=1}^{N_{ps}} \left(\frac{\partial D}{\partial I_n}\right)^2 \sigma_{I_n}^2} \tag{S32}$$

where the partial differential terms can be derived by Eqs. (S25) and (S26) and σ_{I_n} is the image noise during the capture. In the experiments, the tested image noise of our camera σ_{I_n} is 2.4. The bin interval Δp was set as 0.04. Figure S4 shows the predicted numerator and the denominator of the BNN for the test case where both the training data and the testing data are $f = 160$ fringe patterns. It can be seen that the uncertainty maps can successfully indicate the errors of the numerator and the denominator. From the reliability diagrams, although the model shows a little overconfidence, the whole predicted credibility generally matches the actual accuracy, leading to a curve close to $y = x$ and indicating that the proposed model has been well calibrated.

Further, to test the accuracy of the predicted uncertainty maps for the out-of-distribution (OOD) scenarios, we analyzed the estimated numerator and denominator in the case where the BNN was trained with fringe patterns of projected frequency $f = 160$ but was tested with fringe patterns of $f = 80$. The results are shown in Fig. S5, in which the error has been increased significantly and this problem was captured successfully by our uncertainty maps. From the corresponding reliability diagrams, we can see that the model has also been well calibrated in this case.

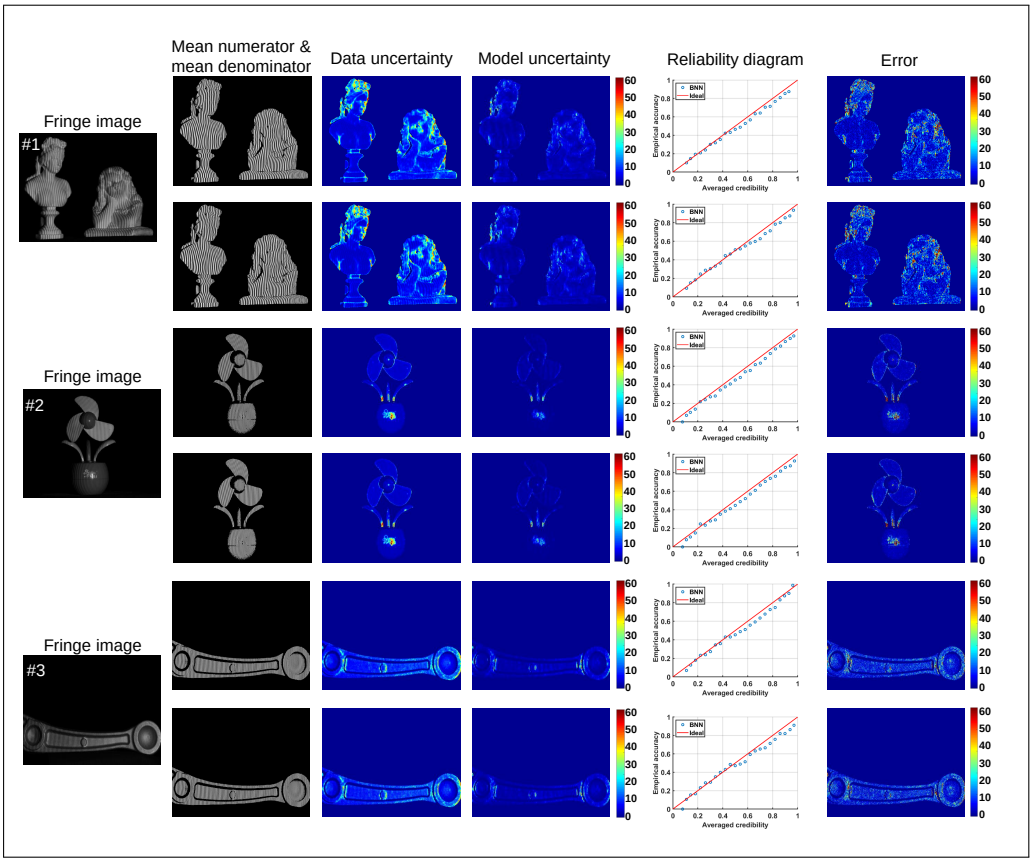

Fig. S4. Analysis of the estimated uncertainty for the numerator and the denominator of in-distribution testing data. In each scene, the first row corresponds to the results of the numerator, and the second row corresponds to the results of the denominator. The reliability diagrams indicate a well-calibrated BNN.

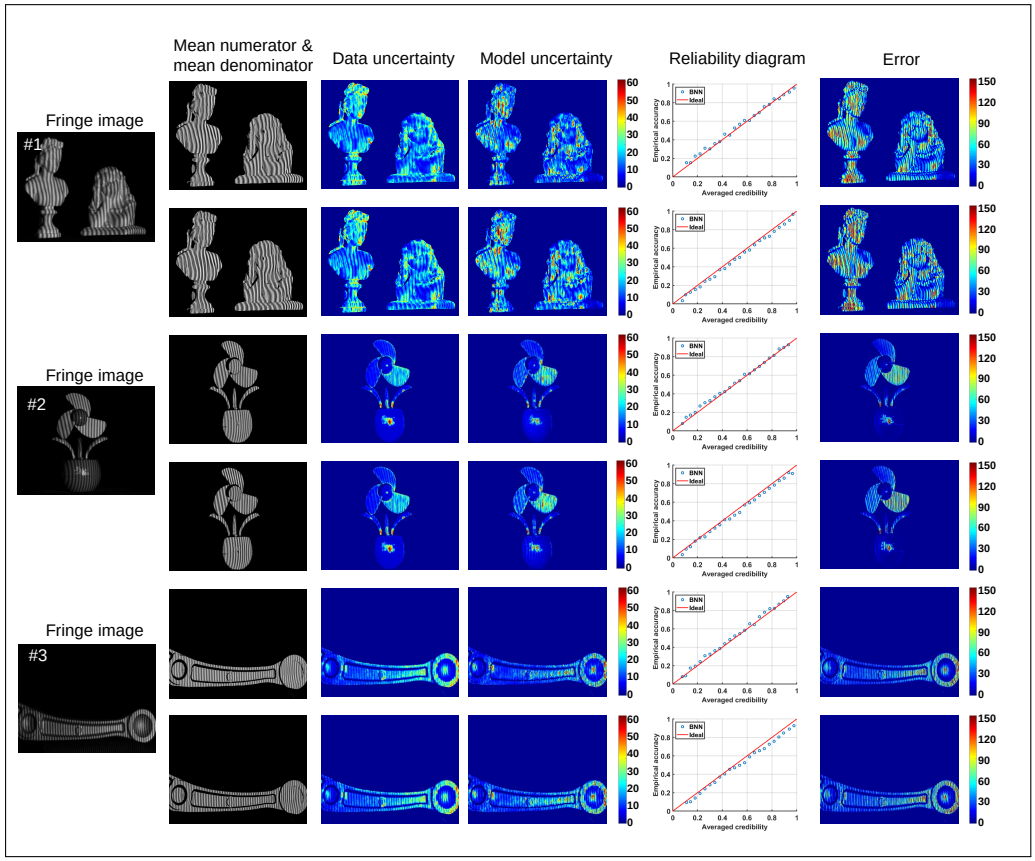

Fig. S5. Analysis of the estimated uncertainty for the numerator and the denominator of out-of-distribution testing data. In each scene, the first row corresponds to the results of the numerator, and the second row corresponds to the results of the denominator. The error increases significantly and the uncertainty maps faithfully captured this issue. From the reliability diagrams, the BNN has also been well calibrated in this case.

5. ANALYSIS OF FRINGE PATTERNS IN UNSEEN IMAGE DOMAINS

More never-experienced (i.e., out-of-distribution) input data were tested to show the effectiveness of the proposed method. In the first experiment, we trained the BNN with fringe images of spatial frequency of $f = 160$ but tested it with fringe images of a different spatial frequency, i.e., $f = 80$. The first scene consisted of two plaster models. The second scene was a plastic desktop fan and the third an industrial part made of aluminum alloy. The first column of Fig. S6 demonstrates the scenes captured with $f = 80$ projected gratings, which appear wider than the fringes of $f = 160$ as shown in the primary manuscript. The second column of Fig. S6 shows the wrapped phase of BNN. After phase unwrapping, we calculated the absolute phase error of the BNN, and the results are shown in the third column of Fig. S6. It can be seen that all the test scenes were measured with heavy phase errors, which have been captured faithfully by both the data uncertainty and the model uncertainty, as shown in the last two columns of Fig. S6.

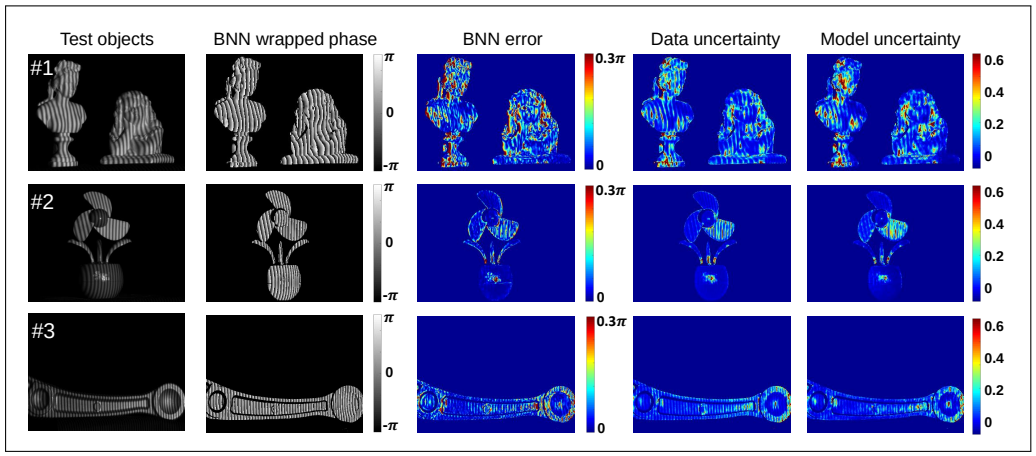

Fig. S6. Experimental results when the BNN was trained with fringe images of spatial frequency of $f = 160$ but was tested with fringe images of $f = 80$. The first column shows the input fringe image. The second and the third show the wrapped phase and the phase error of the BNN. The last two columns demonstrate the data uncertainty and the model uncertainty of the predicted phase respectively.

In Table S1, we calculated the mean absolute error(MAE) and mean uncertainties for quantitative comparison. It can be seen that the phase quality of BNN was severely degraded, since these models never saw the fringes of $f = 80$ in the training stage. Compared with the errors shown in Table S2 where both the training data and the testing data were captured from the projected fringes of $f = 160$, the MAE of BNN here almost tripled or even quadrupled. Our BNN also captured this problem, giving almost significant increasing uncertainties. Further, we find that the change of the model uncertainty is more obvious than that of the data uncertainty, indicating the model's sharply rising uncertainty about its predictions.

Table S1. Quantitative analysis of the BNN when the testing data were captured from a different image domain that consisted of fringes of $f = 80$.

	#1	#2	#3
MAE of BNN (rad)	0.34	0.19	0.24
Mean data uncertainty (rad)	0.17	0.10	0.13
Mean model uncertainty (rad)	0.18	0.10	0.12

In the second experiment, we kept the training data unchanged but further modified the testing dataset, which purely consisted of background images without any stripe structures. The input images are shown in the first column of Fig. S7. The wrapped phase of BNN is demonstrated

Table S2. Quantitative analysis of the BNN when both the training data and the testing data were captured from the projected fringes of $f = 160$.

	#1	#2	#3
MAE of BNN (rad)	0.072	0.071	0.066
Mean data uncertainty (rad)	0.073	0.072	0.059
Mean model uncertainty (rad)	0.029	0.026	0.024

in the second column. It can be seen that the predicted phase maps were terrible. As there is no stripe structures in the input image, the outputs were far from normal wrapped phase maps. The last two column of Fig. S7 show the uncertainty maps of the BNN, both of which have indicated serious errors for these scenes. By calculating the mean uncertainty as shown in Table S3 , we find that the second scene shows that largest mean data and model uncertainties that are 1.37 rad and 1.24 rad respectively while the first scene has the relatively small mean uncertainty that is still around 0.7 rad. From these experiments, we can see that the phase error tends to increase significantly when the testing image is extracted from a different image domain and the uncertainty maps given by our BNN can faithfully predicted the corresponding error distribution.

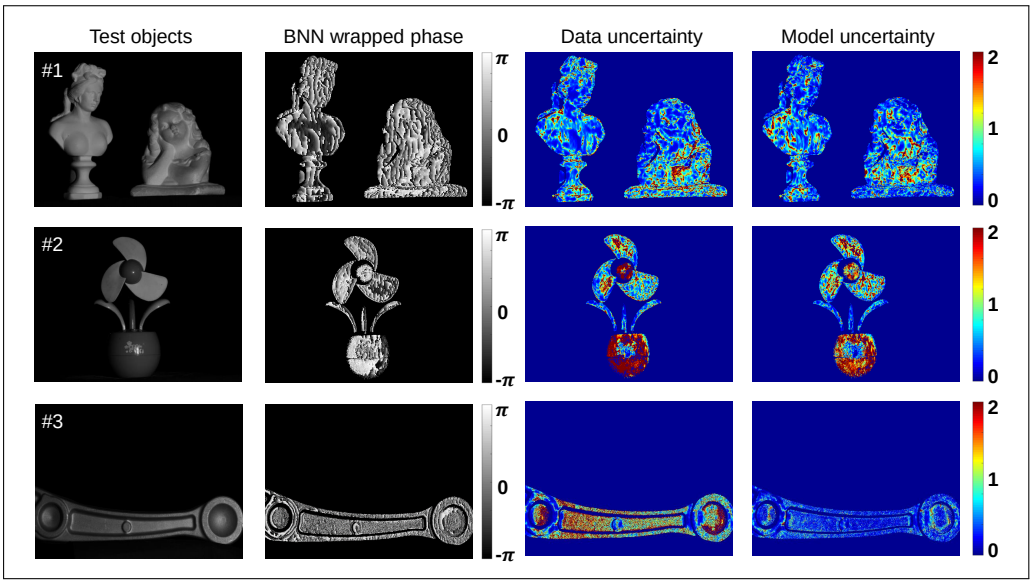

Fig. S7. Experimental results when the BNN was trained with fringe images of spatial frequency of $f = 160$ but was tested with background images that were captured without projected fringes. The first column shows the test scenes. The second column demonstrates the wrapped phase predicted by BNN. The last two column show the data uncertainty and the model uncertainty predicted by BNN respectively.

Table S3. Mean uncertainties of the BNN that was trained with fringe images of spatial frequency of $f = 160$ but was tested with background images that were captured without projected fringes.

	#1	#2	#3
Mean data uncertainty (rad)	0.76	1.37	1.23
Mean model uncertainty (rad)	0.69	1.24	0.52

6. PROCESSING FRINGE PATTERNS OF DIFFERENT SPATIAL FREQUENCIES

Here, we tested the BNN with fringes of different spatial frequencies. The selected spatial frequency includes $f = 40$, $f = 80$, $f = 160$, and $f = 200$. Fringe patterns with the same frequency form a group of training and testing dataset. Then, we trained and tested our BNN with the fringe patterns that have the same spatial frequency. Figures S8, S9, S10, and S11 demonstrate the phase error and the uncertainty maps of the BNN in these cases. We can see that when the frequency is 200, the phase error of the smooth region is relatively small. The large error can be found at the areas with complex contour, such as the hair of the two models and the edges of these objects. Also, we find that there are highlight regions in the second scene. Since the pixels in the highlight region are saturated, it is difficult for the BNN to recover the phase information of these regions accurately. We can see that the areas with large errors are labeled with large data uncertainty and model uncertainty by the neural network. When the spatial frequency reduced to 160, the surface with complex shapes still shows relatively large errors, which are also indicated by the predicted uncertainty successfully. We can see that the measurement error increases obvious when the frequency decreases to 40. Some smoother areas begin to have increasing errors and the uncertainty maps have accurately captured this issue.

Figure S12 summarizes the BNN's mean absolute error, mean data uncertainty, and mean model uncertainty. Figure S12(a) shows the MAE of BNN. We can see that for our system when the projection frequencies is 80 or 160, the measurement error of BNN is relatively small. When the fringe becomes denser or sparser, the error begins to increase. The reason may be the fact that when the fringes are dense, the modulation of the grating tends to decrease, thus leading to larger errors. On contrary, if the fringe is too wide, most of the useful information would be easily hidden in the wide and dark stripe, make the BNN hard to perceive. Figures S12(b) and (c) show the measurement uncertainty of the phase. It can be found that for each tested scene, when the BNN error increases, the corresponding data uncertainty and model uncertainty increase accordingly. As this work only focuses on the uncertainty estimation of the neural network's output, the BNN trained by the $f = 160$ fringe patterns can used to validate the proposed method. We tend to further investigate how to retrieve the phase accurately from quite dense or sparse fringe patterns in the future.

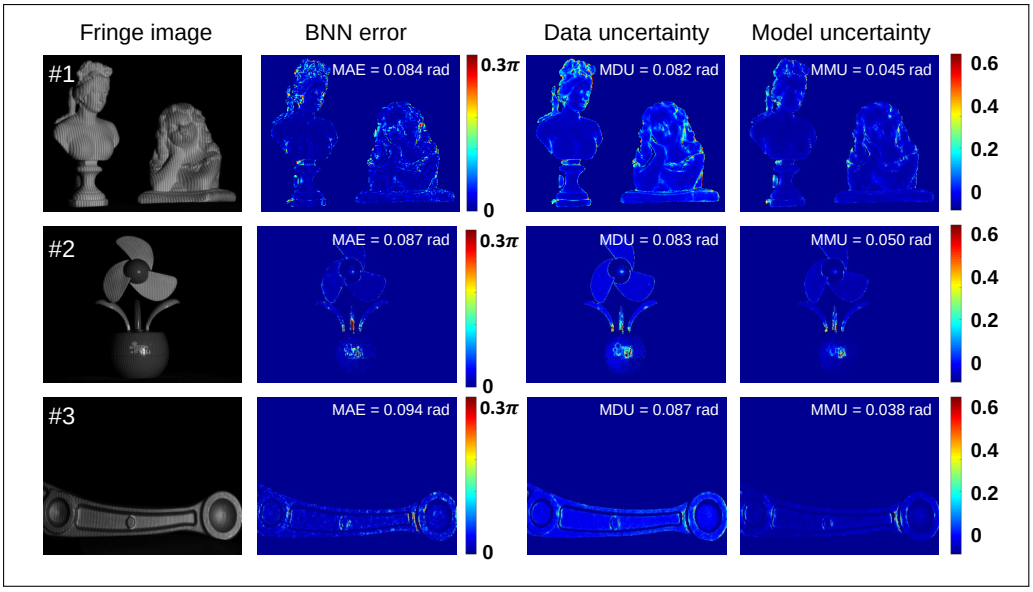

Fig. S8. Analysis of fringe patterns with $f = 200$. Three scenes were tested. The input fringe image to the BNN is shown in the first column. The second column demonstrates the absolute phase error of BNN (MAE: mean absolute error). The third and the fourth column shows the data uncertainty (MDU: mean data uncertainty) and the model uncertainty (MMU: mean model uncertainty) of predicted phase.

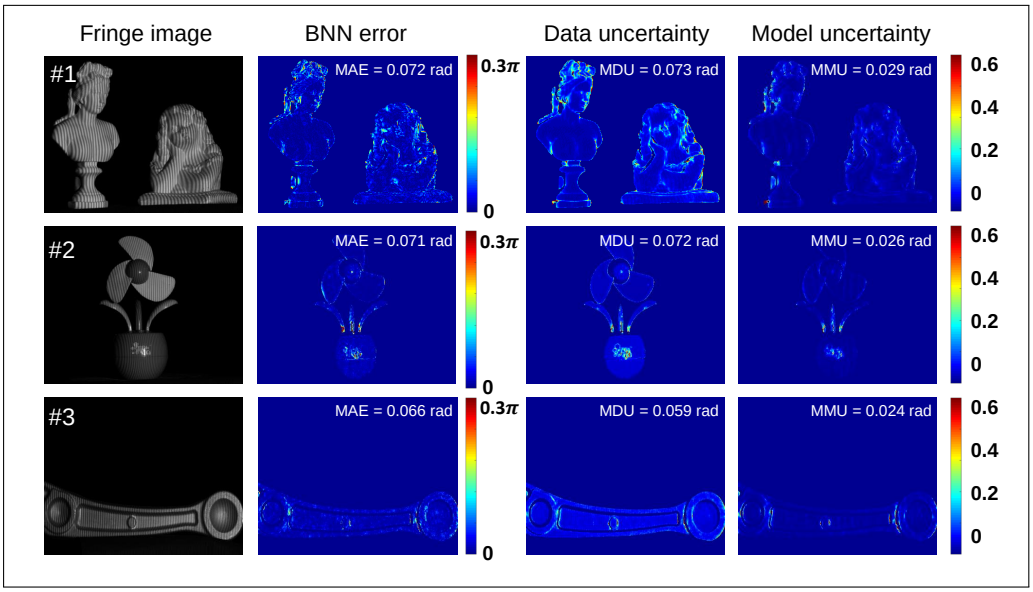

Fig. S9. Analysis of fringe patterns with $f = 160$. Three scenes were tested. The input fringe image to the BNN is shown in the first column. The second column demonstrates the absolute phase error of BNN (MAE: mean absolute error). The third and the fourth column shows the data uncertainty (MDU: mean data uncertainty) and the model uncertainty (MMU: mean model uncertainty) of predicted phase.

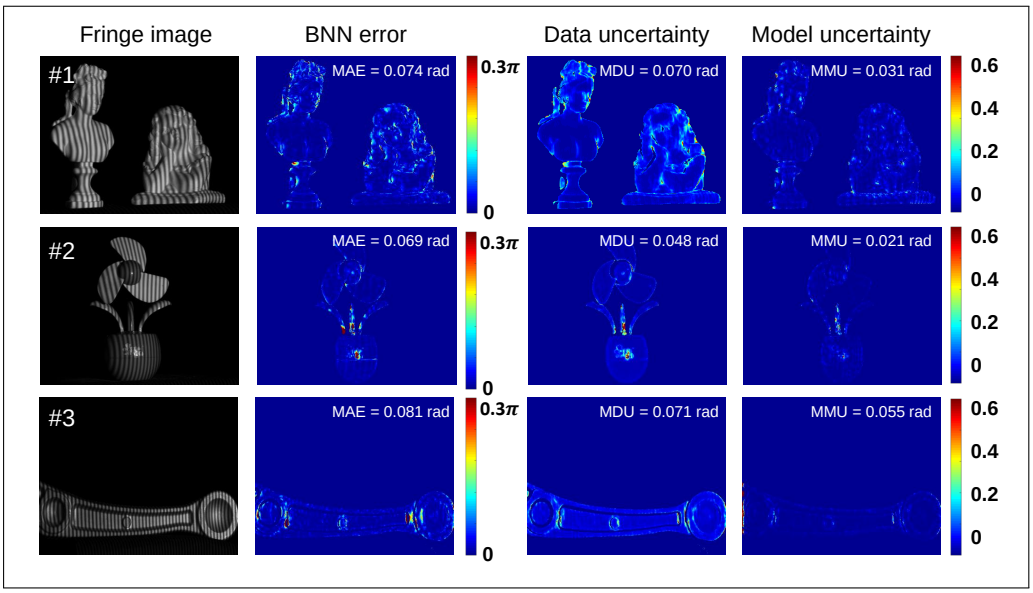

Fig. S10. Analysis of fringe patterns with $f = 80$. Three scenes were tested. The input fringe image to the BNN is shown in the first column. The second column demonstrates the absolute phase error of BNN (MAE: mean absolute error). The third and the fourth column shows the data uncertainty (MDU: mean data uncertainty) and the model uncertainty (MMU: mean model uncertainty) of predicted phase.

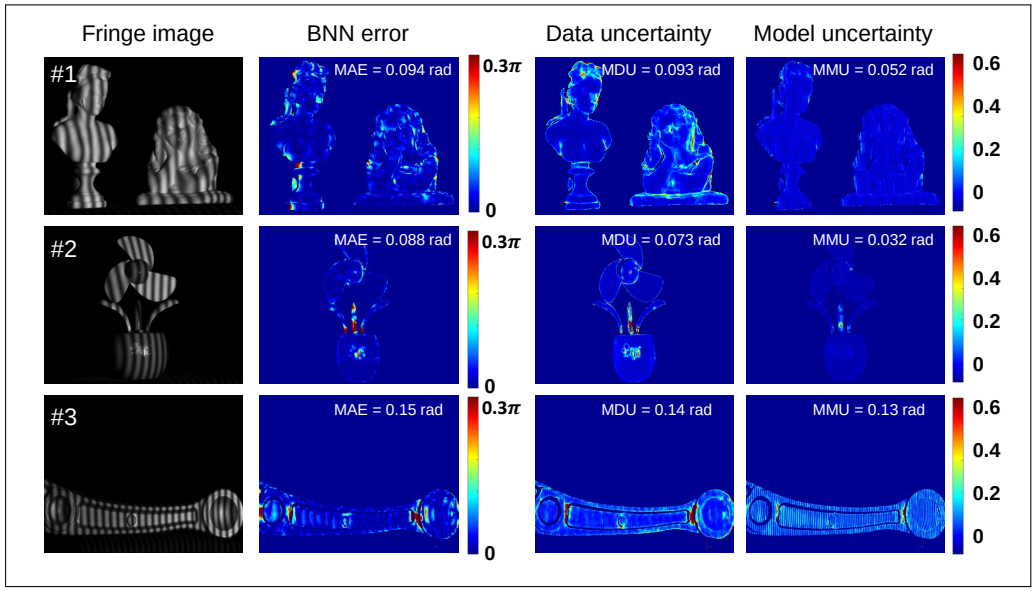

Fig. S11. Analysis of fringe patterns with $f = 40$. Three scenes were tested. The input fringe image to the BNN is shown in the first column. The second column demonstrates the absolute phase error of BNN (MAE: mean absolute error). The third and the fourth column shows the data uncertainty (MDU: mean data uncertainty) and the model uncertainty (MMU: mean model uncertainty) of predicted phase.

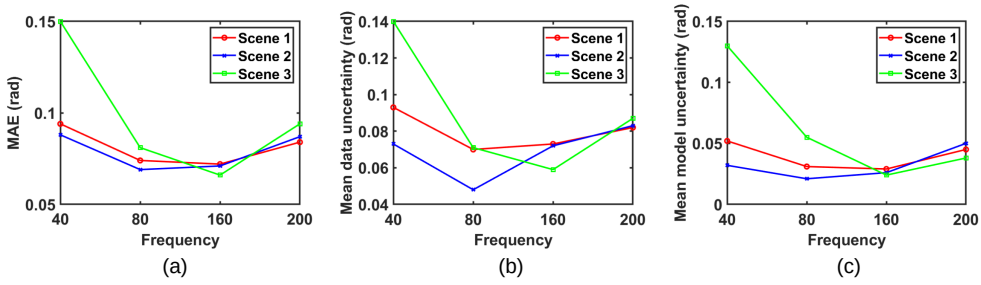

Fig. S12. BNN's performance for fringe pattern analysis with different spatial frequencies. (a) The mean absolute error (MAE) of BNN. (b) The predicted mean data uncertainty. (c)The predicted mean model uncertainty.

7. DROPOUT RATES LEARNED BY BNN

In the theory of dropout, the neurons of a certain layer are set inactive according to the dropout rate. The corresponding probability distribution is Bernoulli distribution. In this work, the dropout rate of each layer (22 layers in our BNN) can be learned by BNN during the training. In each layer of BNN, the dropout rate was initialized randomly according to a uniform distribution $[0.2, 0.6]$ and then it was updated within the range $[0.0, 1.0]$ during the training process. We trained the BNN with fringe patterns of different frequencies and the learned dropout rates are shown in Fig. S13. For all these cases, we can see that the overall distribution of the dropout rate is like a Gaussian distribution. To be specific, in the cases of $f = 200$, $f = 160$, $f = 80$ the first several layers and the last several layers nearly do not drop neurons while the middle layers tend to drop near 50% neurons. The reason may be the fact that the basic architecture of our BNN is the U-Net which has a symmetrical structure. The BNN learns to use almost all the neurons so as not to miss pixel-level information in some of the first layers and the last layers (which can observe the low-level information owing to the concatenation and upsampling processes in the U-Net). While, the middle layers handle the high-level information that is extracted from the low-level information. It is advantageous to drop a part of neurons to avoid overfitting. We find that the learned dropout rates are very small when frequency reduces to $f = 40$. It indicates that nearly all neurons of the BNN are used. Actually, the BNN error of $f = 40$ is relatively larger than that of $f = 80$, $f = 160$, $f = 200$ as can be seen in Fig. S12(a). It may imply that more powerful networks with increasing neurons may be necessary to process the sparse fringe patterns. We will further investigate methods that can handle sparse patterns in the future.

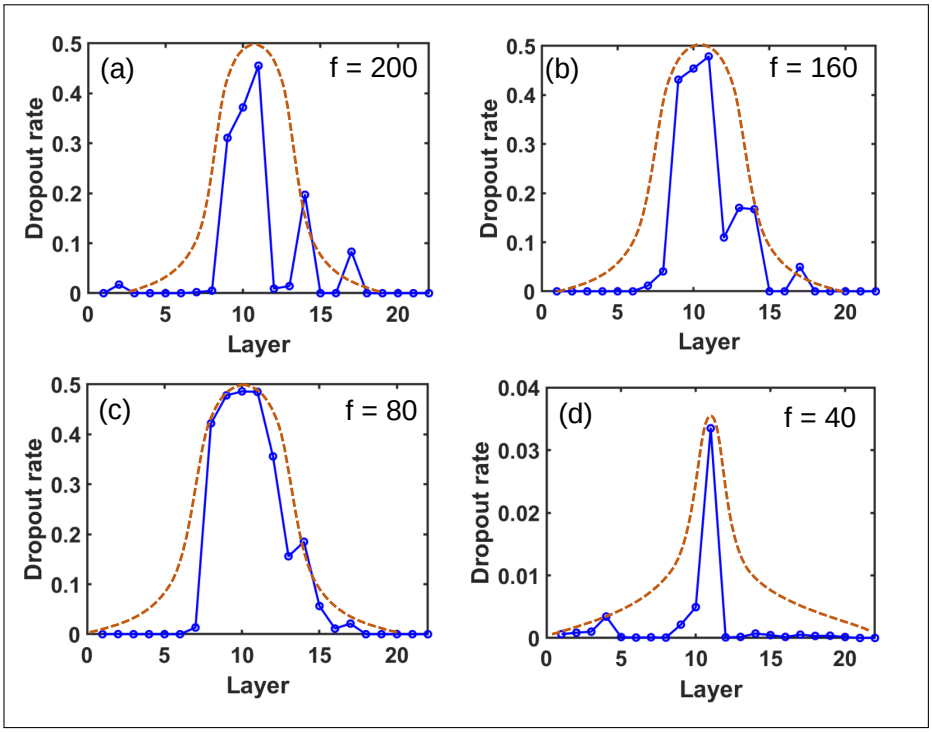

Fig. S13. The dropout rates of each layer learned by the BNN. (a) The BNN trained with fringe patterns of $f = 200$. (b) The BNN trained with fringe patterns of $f = 160$. (c) The BNN trained with fringe patterns of $f = 80$. (d) The BNN trained with fringe patterns of $f = 40$. The solid line shows the actual learned dropout rate and the dashed line represents the overall trend of the varying dropout rate.

8. 3D RECONSTRUCTIONS OF BNN

As it may be difficult to observe the retrieved details from the recovered phase, we converted the unwrapped phase into 3D reconstructions as shown in Fig. S14. For comparison, we also

used the widely applied Fourier transform profilometry(FTP) to analysis the single input fringe image [6] and used the 12-step phase-shifting (PS) algorithm to obtain the ground-truth 3D reconstruction. We can see that it is difficult for FTP to accurately reconstruct the surfaces with complex shapes, e.g., the hairs of both models in the first scene. For our method, however, it can faithfully reconstruct the 3D surfaces for both complex details and smooth areas. Further, the BNN only used a single fringe image and almost reproduced the ground-truth 3D results that were obtained with 12 PS images.

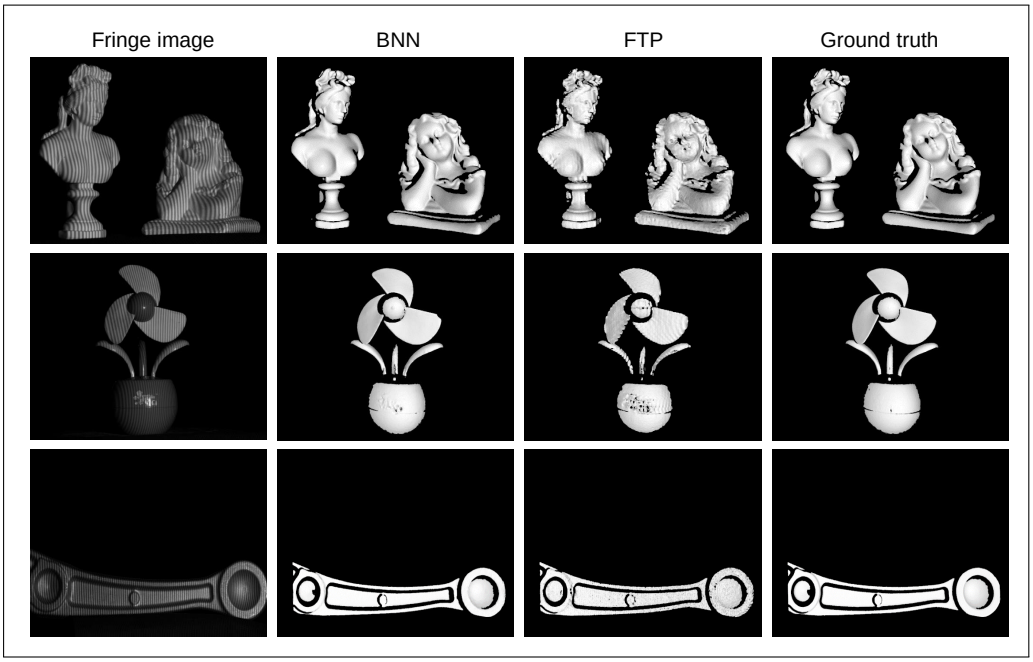

Fig. S14. Comparison of the 3D reconstruction obtained with different methods. The first column shows the test fringe image with projected frequency of $f = 160$. The second column the 3D models of BNN. The third column the 3D models of FTP. The last column the ground-truth 3D models.

REFERENCES

1. Y. Gal, J. Hron, and A. Kendall, "Concrete dropout," arXiv preprint arXiv:1705.07832 (2017).
2. O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention,* (Springer, 2015), pp. 234–241.
3. D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980 (2014).
4. C. Zuo, S. Feng, L. Huang, T. Tao, W. Yin, and Q. Chen, "Phase shifting algorithms for fringe projection profilometry: A review," Opt. Lasers Eng. **109**, 23–59 (2018).
5. Y. Xue, S. Cheng, Y. Li, and L. Tian, "Reliable deep-learning-based phase imaging with uncertainty quantification," Optica **6**, 618–629 (2019).
6. M. Takeda and K. Mutoh, "Fourier transform profilometry for the automatic measurement of 3-D object shapes," Appl. optics **22**, 3977–3982 (1983).