Optics EXPRESS

Phase unwrapping using deep learning in holographic tomography

MICHAŁ GONTARZ,[*] VIBEKANANDA DUTTA, MAŁGORZATA KUJAWIŃSKA, AND WOJCIECH KRAUZE

Warsaw University of Technology, Institute of Micromechanics and Photonics, 8 Boboli St., 02-525, Warsaw, Poland
[*]*michal.gontarz@pw.edu.pl*

Abstract: Holographic tomography (HT) is a measurement technique that generates phase images, often containing high noise levels and irregularities. Due to the nature of phase retrieval algorithms within the HT data processing, the phase has to be unwrapped before tomographic reconstruction. Conventional algorithms lack noise robustness, reliability, speed, and possible automation. In order to address these problems, this work proposes a convolutional neural network based pipeline consisting of two steps: denoising and unwrapping. Both steps are carried out under the umbrella of a U-Net architecture; however, unwrapping is aided by introducing Attention Gates (AG) and Residual Blocks (RB) to the architecture. Through the experiments, the proposed pipeline makes possible the phase unwrapping of highly irregular, noisy, and complex experimental phase images captured in HT. This work proposes phase unwrapping carried out by segmentation with a U-Net network, that is aided by a pre-processing denoising step. It also discusses the implementation of the AGs and RBs in an ablation study. What is more, this is the first deep learning based solution that is trained solely on real images acquired with HT.

1. Introduction

Quantitative Phase Imaging (QPI) describes a vast array of State-of-the-Art phase measurement techniques, constantly expanding and improving [1,2]. It enables high-quality measurements of transparent objects, whose effect on phase delay is directly represented in the results. Crucially, unlike fluorescence microscopy, it does not require labeling, reducing the risk of photobleaching or phototoxicity while improving the measurement speed and cost.

Holographic tomography (HT) is one of the most popular 3D QPI techniques used when a high-resolution reconstruction of a specimen's inner structure is needed [1]. The 3D refractive index (RI) distribution is provided by tomographic reconstruction based on complex amplitudes retrieved from the multiplicity of digital hologram projections of an object illuminated from different directions. Due to the nature of phase coding in holograms or interferograms, the phase retrieved from the fringe patterns is in the form of $mod2\pi$, i.e., it exhibits discontinuities and requires further processing through a phase unwrapping (PU) procedure. PU is the most challenging and demanding procedure during the final phase retrieval process in the presence of high noise and domain discontinuities. In the case of HT, failures of PU for even a few holographic projections may induce significant errors in the reconstructed RI, so it is essential to ensure its reliability and high performance.

Although PU has been tackled during the last thirty years by many research groups and numerous path-dependent [3–6], and path-independent [7–9] methods have been developed, they lack, in difficult cases, in noise robustness, reliability, speed, and automation.

With the late development of artificial intelligence (AI) and deep learning (DL), PU has been approached using convolutional neural networks (CNNs), which have proven their significant contribution in pattern recognition tasks [10]. By adjusting the weights of a convolution kernel in the networks, the model studies the image's features and their relations. This ability means

#486984 https://doi.org/10.1364/OE.486984
Journal © 2023 Received 8 Feb 2023; revised 27 Mar 2023; accepted 29 Mar 2023; published 23 May 2023

that, nowadays, DL is a State-of-the-Art solution of image classification [11–13], segmentation [14–16], object detection [17,18], image denoising [19–22] and PU [23–27] problems. In particular, PU has been solved in two ways: semantic segmentation (SS) and image translation (IT). SS labels pixels with integers according to their fringe order, generating a wrap count map. It is then multiplied by 2π and added to the wrapped phase image, generating a continuous phase distribution. It establishes a direct relationship between the wrapped phase input and the output, transforming the image from one domain to another.

This paper presents a DL-based pipeline capable of denoising of wrapped phase images and unwrapping the ($mod2\pi$) phase. The denoising and unwrapping algorithms work under the umbrella of CNNs. These two models work in tandem as a pipeline, with the output of the denoising step being the direct input to the unwrapping model. Synthetically generated dataset with varying complexity and noise has been used as training data for the denoising model, which preserves the essential information at the discontinuities of the object. The unwrapping model incorporates a U-Net [28] backbone, however, it utilizes Attention Gates (AG) [29] and Residual Blocks (RB) [12] and, unlike other State-of-the-Art DL-based PU solutions, has been trained on a large dataset of real data acquired in HT. Specimen captured in this dataset contains complex scattering structures, and their measurements exhibit high noise, which results in irregular phase fringes. The pipeline has been evaluated on real data from HT measurements and synthetic data, created in a way that can generate phase images of cell-like structures with aberrations and noise. Note that this paper focuses solely on the analysis of PU by the proposed pipeline, which is why all the results and tests are performed on single projections of HT data, as opposed to reconstructing an entire 3D volume for HT. The proposed pipeline is the first work in which the unwrapping, performed by semantic segmentation, is trained entirely on the images acquired in HT. Other State-of-the-Art models are trained on synthetically generated images or images that do not exhibit irregularities in phase fringes. Therefore we predict, that their performance will not match that of our proposed pipeline with real HT images.

This paper consists of six parts. Firstly, a brief description of the theory of phase retrieval and unwrapping within the scope of HT is presented in Section 2. Section 3 provides detailed insight into the individual models in the pipeline, as well as the chosen hyperparameters and evaluation metrics. The testing stage of model development and its evaluation assessment is presented in Section 4. The results, discussion, and overall performance of the proposed method against the State-of-the-Art techniques are presented in Section 5. Finally, Section 6 concludes and reviews the proposed methodology.

2. Theoretical background

2.1. *Processing pipeline in holographic tomography*

HT [30,31] is a quantitative label-free imaging technique that reconstructs the 3D RI of a transmissive or quasi-transmissive object from multiple holographic projections. The label-free nature makes it attractive for biological analysis, as specimens, like cells and tissues, are inherently semi-transparent. RI distribution and its changes are related to the biophysical features of the specimens and are utilized in such fields as cytometry, cell biology, biotechnology, infectious disease, hematology, and cancer research.

In biomedical applications, the projections are most often captured in a Mach-Zehnder interferometer configuration with the object illumination beam $u_{i\phi}$ scanning (Fig. 1). The beam scattered $u_{s\phi}$ at the object is interfering with the reference beam u_r and creates an in-plane hologram with carrier frequency which is captured by the camera (CAM). As shown in Fig. 1, each hologram is processed by the Fourier transform method [32] with the aim to retrieve an object phase in sequential projections. The retrieved data is in the form of phase fringes ($mod2\pi$), and they require PU. The PU step in the HT processing pipeline (the blue box in Fig. 1) is the point of interest in our paper. After PU, the phase projections are further processed by one of the tomographic reconstruction procedures [33]. Each tomographic measurement requires capturing from a few tens up to a few hundreds of holograms for a single volumetric measurement. If some of the phase projections contain gross errors due to incorrect unwrapping, the final reconstruction of 3D RI is erroneous [34] without the possibility to control or predict the character and value of the error. This statement is general, and it also refers to other HT system configurations and holographic reconstruction procedures and phase retrieval methods such as temporal or spatial phase shifting methods [35]. It should be noticed that recently HT has been applied for investigations of bigger and/or more scattering biological objects, including tissues, organoids, and small organisms [36–38], which provide very noisy and complicated *mod*2π maps. This is the reason why novel approaches to PU are required.

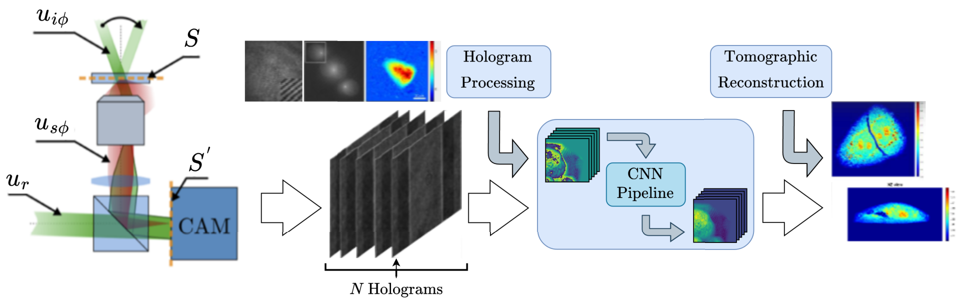

Fig. 1. Schematics of the proposed modification to the measurement pipeline in HT, including the system and the processing.

2.2. Phase unwrapping

The basic PU process includes scanning of phase fringes and, upon encountering a phase discontinuity, adding a multiple of 2π. Since measured phases can exhibit local shadows, irregular shapes, brightness, discontinuities, and noise, sophisticated PU algorithms have been developed to solve this problem with high confidence [39]. Conventional algorithms have been described in Section 2.2.1, while Section 2.2.2 presents the development of different PU algorithms, which are based on DL techniques.

2.2.1. Conventional methods

The conventional methods describe the PU algorithms that have not been developed under the umbrella of AI or DL. Those methods can be divided into two groups: path-dependent, that follow a path on the phase image, and path-independent [35], which tackles the unwrapping problem globally. However, at the moment, global solutions based on regularization are not widely applied. Typically, HT processing uses path-dependent algorithms so that we will focus on them.

The most important algorithm, from the authors perspective, has been developed by Arevallilo-Hernandez et al. [3] and is called Quality Guided Phase Unwrapping (QGPU). It is one of the most robust and popular methods for 2D PU, and serves as a benchmark method in our HT processing algorithm. What is more, this method is used to obtain all experimental ground truth (GT) phase images mentioned in this paper, which is why it will be discussed in detail. This method starts with computing a quality map for the wrapped phase image. Pixels of this map are then connected by horizontal and vertical edges, for which an edge quality is calculated based on the values of the connected pixels. Edges are then sorted in decreasing order by the quality value and queued. Each pixel is assigned to a different group. Iterating over the queue performs further operations. If the pixels connected by the edge belong to the same group, the edge is discarded. Otherwise, the unwrapping operator $\mathcal{U}(x_i, x_j)$ is computed, which determines the integer multiple k of 2π, which, when added, would unwrap the second pixel. Finally, $k2\pi$ is added to all the pixels in the group of the second pixel.

$$\mathcal{U}(x_i, x_j) = 2\pi \left\lfloor \frac{x_i - x_j + \pi}{2\pi} \right\rfloor \tag{1}$$

where x_i and x_j denote the first and second pixel of the edge, whilst $\lfloor . \rfloor$ denotes the *floor* operation [3].

This quality-guided unwrapping method shows excellent results regarding speed and robustness; hence, it is used by many researchers as a go-to State-of-the-Art solution when facing a PU problem. This method has proven the most reliable in our experiments in the past, which is why it has been used as a benchmark for evaluating the DL solution proposed in this article, and we chose it as the GT method for PU comparisons.

Other notable solutions are based on tree-growing strategies and include the Minimal Spanning Tree (MST) [4] and an algorithm proposed by T. J. Flynn in [5], which has been since improved upon by J. Xu et al. in [6].

Nevertheless, the performance of the conventional algorithms can be hindered when the investigated phase images exhibit noise, discontinuities, or high irregularities in the phase fringes. Therefore, the introduction of AI, especially contemporary DL methods, has been widely utilised to tackle this problem.

2.2.2. Deep learning methods

Due to the growing popularity of machine learning domain adaption, several DL techniques have been applied to solve the problem of PU [40]. The CNNs are very popular due to their high performance ability in image processing, whilst limiting computational cost and inference time. They are able to learn features of varying complexity (edges, shapes) about images of different types and modalities. Their name and its operation are based on the convolution operation.

The DL-based phase unwrapping has been tackled by many researchers and through many different approaches. Those involve PU by semantic segmentation [26,41,42]. In [43] Zhang et al. proposed a solution, where PU is done by segmentation, and is aided by a pre-processing DL-based denoising step. Another approach to DL-based PU is image translation, where a direct mapping between wrapped and unwrapped phase distributions is obtained. Wang et al. in [27] proposed the integration of RBs into a U-Net, which provided good and stable results. PhUn-Net is a deep residual network proposed by Dardikman-Yoffe et al. in [44], which is based on the same approach.

Other notable DL-based PU solutions include using a Recurrent Neural Network in [45] and a Generative Adversarial Networks (GANs) [46] in [47] and Pix2Pix GANs in [23]. Yang et al. developed a Deep Image Prior based PU approach in [24], which is a unique DL-based approach, because it does not require a training set. A thorough review of recent DL-based PU developments has been compiled by Wang et al. in [48].

In CNNs, convolution can be defined as

$$I_{\text{out}}[m, n] = \sum_j \sum_i I_{\text{in}}[m + j, n + i] \cdot K[j, i] \tag{2}$$

where m and n denote the coordinates within the image matrix, j and i denote coordinates within the convolution kernel, $I_{\text{out}}, I_{\text{in}}$ describe output and input image matrices respectively, and K denotes the convolution kernel.

PU method proposed in this paper is based on the semantic segmentation task, which is very popular in CNN applications. Semantic segmentation means label assignment for each pixel in an image.

In our application, different labels are assigned to different phase wrap counts, as shown in Fig. 2. Such wrap count is then cast into integer form (e.g., label "0") and multiplied by 2π. Adding the result to the initial wrapped phase image results in an unwrapped phase image, in which errors can only be results of misclassifications of pixels and have values of $k2\pi$, where k is an integer multiple resulting from the segmented label.

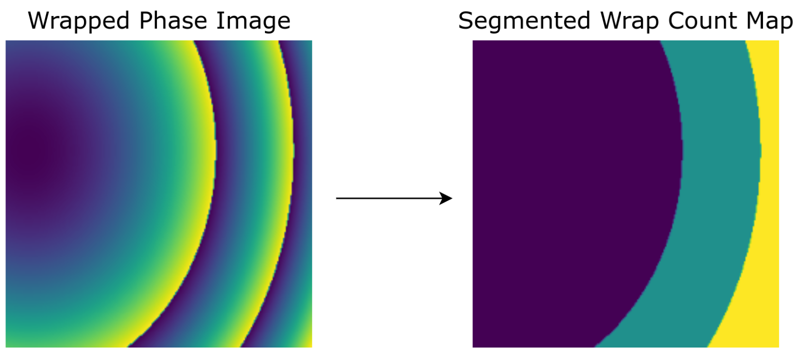

Fig. 2. Example of generating a wrap count map from a wrapped phase image.

This paper focuses on the U-Net [28], an architecture that has seen growing popularity in this area. Its energy function $E_{\text{U-Net}}$ [49] can be defined as a mathematical combination of a 'softmax' activation function f_{softmax} on the final feature map combined with a cross-entropy loss [28,49]. Softmax is a combination of multiple sigmoid functions f_{sigmoid} [50]. It returns probabilities of the predicted pixel being labeled to a particular class.

$$E_{\text{U-Net}} = \sum_{x\in\Omega} w(x) \log[p_{\ell(x)}(x)] \tag{3}$$

where ℓ denotes the true label values and w is a weight map 'learned' during training.

$$f_{\text{softmax}}(z)_j = \frac{e^z_j}{\sum_{k=1}^K e^z_k} \quad \text{for } j = 1, \ldots, K. \tag{4}$$

$$f_{\text{sigmoid}}(x) = \frac{1}{1+e^{-x}} \tag{5}$$

This architecture's name originates from its unique 'U-shaped' architecture, which can be divided into three parts: encoding (contracting) path, decoding (expanding) path, and the bottleneck (bridge) that connects the two. The encoding path's job is extracting features from an image with the convolution operation with an arbitrary kernel size [28]. The decoding path is responsible for upsampling the feature maps. Each level of this part starts with an upsampling operation, increasing the feature map dimensions by a factor of two. It is followed by a convolution operation with a 2×2 kernel size.

The unique part of U-Nets is the presence of skip connections, which transfer information about feature maps to the decoding path at the same depth. Upsampled features are concatenated with the features transferred through those connections, which is followed by two pairs of convolutions and activation functions. The final layer is convolved through several 1×1 kernels, which corresponds to the number of labels in the segmentation problem [28] and is of shape $N \times M \times k$, where k corresponds to the number of predicted classes and N and M describe the image size.

The U-Net architecture has also been tested and is well-known in tasks such as denoising. In the case of image translation, this model's output and the training datasets' output are in the form of continuous phase images. A direct relationship between the noisy and denoised phase images is established. The difference in model definitions could be in just the final activation function. Since the output is a denoised image, its shape is $N \times M \times 1$.

The training process of the machine learning algorithms is a minimization process. In each iteration (i.e., epoch), the model is trained with the so-called training dataset (which corresponds to $70 - 80\%$ of the entire available dataset), during which the weights are adjusted. In CNNs, the weights are the values of the kernels in the convolutional layers. Next, the trained model is validated against the validation dataset ($20 - 30\%$). Finally, with each epoch, the overall loss is estimated. The commonly used loss function *Cross-Entropy* is favored in segmentation tasks. It is intended for use when optimizing classification models and can be divided into: a) binary-cross entropy and b) categorical-cross entropy, respectively [51].

However, the preferred loss function in image translation tasks is the Mean Squared Error (MSE) [22,44].

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(t_i - s_i)^2 \tag{6}$$

where N denotes the number of observations, or, as in this case, pixels, t_i is the GT label, and s_i is the network prediction.

3. Methodology

3.1. Proposed pipeline

As mentioned in Section 2.2, PU is strongly disturbed by the presence of noise in phase fringes. Therefore, to minimise the influence of noise we propose the two-step process, which includes denoising wrapped phase images (Section 3.2) and unwrapping phase discontinuities through semantic segmentation (Section 3.3) of phase images in $mod 2\pi$ form, which is shown in Fig. 3.

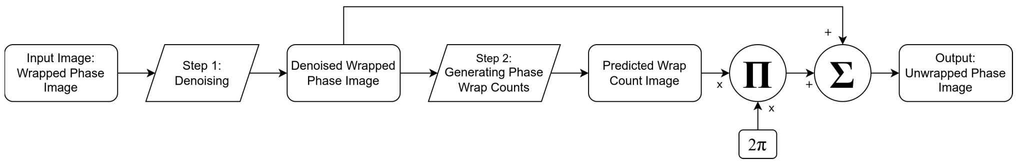
Input Image: Wrapped Phase Image
Step 1: Denoising
Denoised Wrapped Phase Image
Step 2: Generating Phase Wrap Counts
Predicted Wrap Count Image
Π
Σ
Output: Unwrapped Phase Image
2π
x
x
+
+

Fig. 3. Block diagram of the pipeline.

Step 1 of the pipeline processes the input as a translation task. The output is the same wrapped phase distribution, but the noise has been removed. The noise removal model outputs an intermediate image, which is the unwrapping models' input. This process can be described as a translation task. The name of this task has been assigned by the authors of this paper, and it originates from the fact that the task of the model is to translate the data from one domain to another (noised image to denoised image, wrapped phase to unwrapped phase, etc.). Unwrapper outputs the segmented prediction (as in Fig. 2), known as the wrap count. The wrap count map is then multiplied by the value of the discontinuity - 2π, so that the pixels in an area identified as having no wraps will equal to 0, one wrap to 2π, two wraps to 4π, etc. Element-wise addition of this matrix and wrapped phase image results in a continuous phase distribution. The networks' architectures are described in-depth in Sections 3.2 for denoising and 3.3 for unwrapping. Note that, all tensor sizes in the figures (Fig. 5 and Fig. 7) are written, for simplification, assuming input image size of $256 \times 256 \times 1$. In reality, the input image size can be arbitrary, however, the model requires the input data aspect ratio of 1.

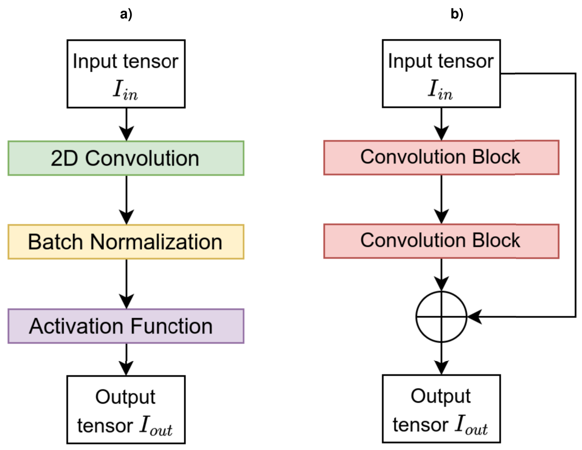

Fig. 4. a) Architecture of a defined convolution block, b) Architecture of a residual block [12].

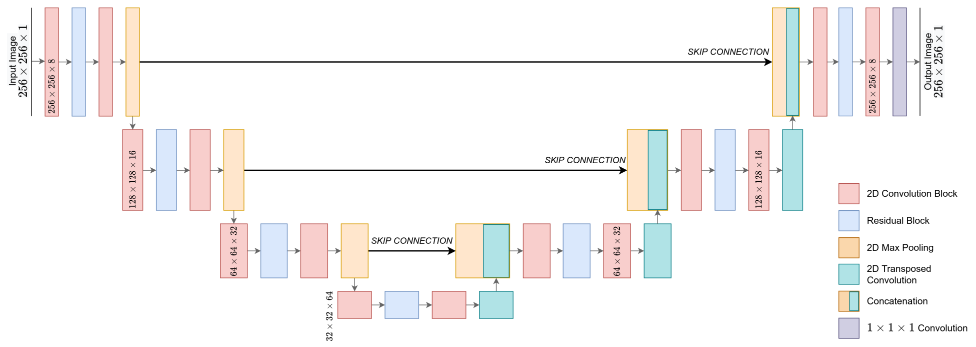
Input Image $256 \times 256 \times 1$
$256 \times 256 \times 8$
SKIP CONNECTION
$256 \times 256 \times 8$
Output Image $256 \times 256 \times 1$
$128 \times 128 \times 16$
SKIP CONNECTION
$128 \times 128 \times 16$
$64 \times 64 \times 32$
SKIP CONNECTION
$64 \times 64 \times 32$
$32 \times 32 \times 64$
2D Convolution Block
Residual Block
2D Max Pooling
2D Transposed Convolution
Concatenation
$1 \times 1 \times 1$ Convolution

Fig. 5. Architecture of the U-Net model used in the noise removal step. The convolutional block is shown in Fig. 4(a). Residual blocks have been defined as shown in Fig. 4(b).

3.2. Noise removal step

Noise removal is implemented in this pipeline as a pre-processing step, during which significant noise is removed, but, most importantly, information at the discontinuities is preserved, improving the performance of the following unwrapping step.

The model responsible for removing the noise of a wrapped image is based on a U-Net [28] backbone architecture. It can be characterized as a 3-level U-Net, because the encoding and decoding paths consist of three levels, with the bottleneck connecting them at the lowest level. The number of filters in the first convolutional operation in the model is a hyper-parameter of the architecture. In this model, it was set to 8. This model uses a non-linear activation function LeakyReLU $f_{\text{LeakyReLU}}$ (leaky rectified linear unit) [52].

$$f_{\text{LeakyReLU}}(x) = \begin{cases} x \ \text{ if } \ x \geqslant 0 \\ \frac{x}{a} \ \text{ if } \ x < 0, \end{cases} \quad \text{where } a \in (1, +\infty) \tag{7}$$

This function is a modification of a ReLU (rectified linear unit) activation function, which *"leaks"* negative values, but multiplies them with the parameter a, whose value was set to 100. The processing path on each level of the decoding path is presented in Table 1. Each level is finalized with a MaxPooling operation (kernel size 3×3, stride (1, 1) and 'same' padding), which halves the feature map size and increases the 'depth' of the model.

Table 1. Stack of consecutive operations on one level of depth in the noise removal U-Net.

Operation	Kernel Size	Stride	Padding
2D Convolution + BN + ReLU	3×3	(1,1)	'same'
Residual Block	3×3	(1,1)	'same'
2D Convolution + BN + ReLU	3×3	(1,1)	'same'

Residual blocks were introduced in [12]. Their architecture can be seen in Fig. 5. Through the use of skip connections and addition of the input to the output of the weighted operation (i.e., convolution), the network does not focus on learning the output $\mathcal{H}(x)$ but instead learns the relation $\mathcal{F}(x) := \mathcal{H}(x) - x$. The learned quantity is the residual between the input and the output, which enables training deeper models with little to no increase in error [12]. Overfitting is tackled by applying Batch Normalization [53].

The bottleneck layer consists of the same operation as on each level on the encoder path, however, at this depth, the number of the feature maps is 2^6 (2^3 times larger than at the first depth level of the U-Net), and their dimensions are 2^3 times smaller than those of the input image.

The decoding layer consists of a stack of the same operations as in the encoding path and the bottleneck, however, it is preceded by a 2D Transposed Convolution operation and concatenation with feature maps transferred through a skip connection at the corresponding depth. As mentioned in Section 2.2.2 the decoding layer upsamples the feature maps at each depth level of the model, by broadcasting the input data via kernel, which means that this is a trainable operation. The number of filters in this operation at depth n is halved, whilst the feature map dimensions are doubled with respect to the feature maps at level $n - 1$. At the shallowest level, the model is finished by a 2D Convolution operation with one 1×1 kernel followed by a LeakyReLU activation function, which means that the final output image's shape is the same as the model's input image shape.

The breakdown of parameters has been depicted in Table 10 in Appendix A. The total number of parameters in the model was 714,185 (712,777 trainable and 1,408 non-trainable parameters).

3.3. Phase unwrapping step

In the context of wrap count map generation, PU is the assignment of an integer value to a pixel, which corresponds to their fringe order as shown in Fig. 2. This results in differences in the output. Its shape has to correspond to the number of classes, that the model is expected to predict, and the final activation function must output values in the range (0; 1], since it determines the probabilities of pixel-wise label assignment.

Whilst the unwrapping model is based on the same backbone as the noise removal model (see Section 3.2), it is more complex and deeper. Whilst the denoising model has 3 levels of depth, the unwrapper has 5 levels of depth. It uses a non-linear activation function ReLU f_{ReLU} [54] (rectified linear unit). It is useful, when tackling overfitting problems. The encoding path can recognize more complex and smaller features of the image, thus performing well on phase images with small and more convex irregularities in phase fringes.

$$f_{\text{ReLU}}(x) = \begin{cases} x \text{ if } x \geqslant 0 \\ 0 \text{ if } x < 0, \end{cases} \tag{8}$$

The model also differs in the skip connection formulation, where AGs have been introduced. Semantic and instance segmentation tasks have benefited from the introduction of AG mechanisms

(Fig. 6) in 2018 by Oktay et al. [29], which uses additive attention q^l_{att}, since it has been proven to, whilst being computationally expensive, provide better results on higher dimensional data [55].

$$q^l_{\mathrm{att}} = \psi^T[\sigma_1(W_x^T x_i^l + W_g^T g_i + b_g)] + b_\psi, \tag{9}$$

$$\alpha_i^l = f_{\mathrm{sigmoid}}[q^l_{\mathrm{att}}(x_i^l, g_i; \Theta_{\mathrm{att}})], \tag{10}$$

where f_{sigmoid} denotes a sigmoid function (Eq. (5)), Θ_{att} denotes a set of parameters, which include: linear transformations W_x, W_g and ψ. Variables x and g denote input pixel and gating signal vectors, respectively [29].

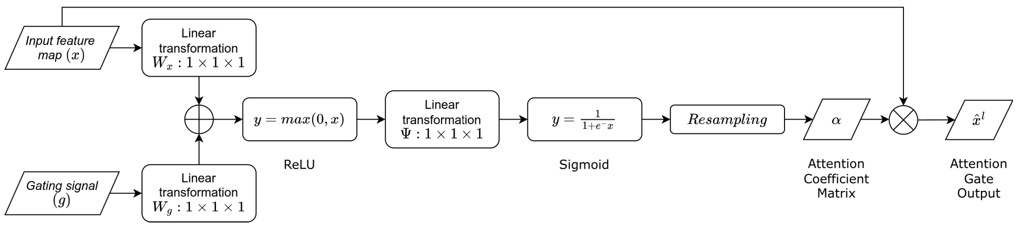
Input feature map (x)
Linear transformation $W_x : 1 \times 1 \times 1$
Gating signal (g)
Linear transformation $W_g : 1 \times 1 \times 1$
$y = max(0, x)$
ReLU
Linear transformation $\Psi : 1 \times 1 \times 1$
$y = \frac{1}{1+e^{-x}}$
Sigmoid
Resampling
α
Attention Coefficient Matrix
$\hat{x}^l$
Attention Gate Output

Fig. 6. Schematics of an AG. Its implementation and schematics have been inspired by [29].

It is a trainable mechanism, which works by generating an attention weight matrix on a feature map. Since it incorporates a sigmoid activation function (Eq. (5)), the values of the weights are output in the range (0; 1]. Output feature maps are multiplied by the attention weight matrix, which suppresses unimportant areas, whilst transmitting salient features, it helps the model 'pay attention to what is important to the task'. The order of operations remains the same as in the denoising model (Table 1) with the only difference being a larger kernel size 5×5.

Higher depth of the mode also means that the number and the dimensions of feature maps at the deepest levels has increased. The first convolution operation uses 8 filters, which means that at the shallowest level, the feature tensor shape is equal to $256 \times 256 \times 8$. Because the dimensions

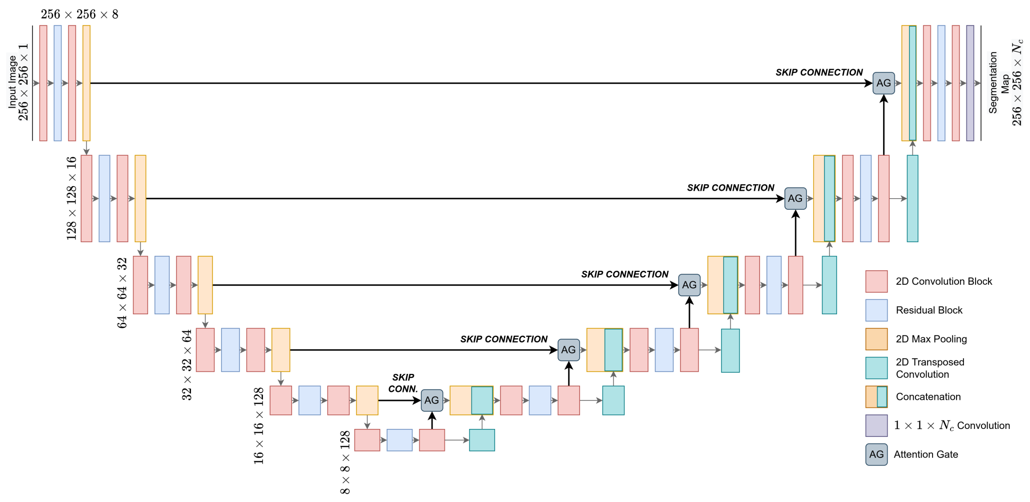
$256 \times 256 \times 8$
Input Image $256 \times 256 \times 1$
$128 \times 128 \times 16$
$64 \times 64 \times 32$
$32 \times 32 \times 64$
$16 \times 16 \times 128$
$8 \times 8 \times 128$
SKIP CONNECTION
SKIP CONN.
AG
Segmentation Map $256 \times 256 \times N_c$
2D Convolution Block
Residual Block
2D Max Pooling
2D Transposed Convolution
Concatenation
$1 \times 1 \times N_c$ Convolution
Attention Gate

Fig. 7. Architecture of the Attention U-Net model used for generating phase wrap-count maps. The final operation in this model is a $1 \times 1 \times N_c$ convolution, followed by a softmax (Eq. (4)) activation function. N_c denotes the number of classes defined in the segmentation task.

are halved and their numbers (as described in Section 2.2.2) are doubled, the feature tensors' dimensions are $8 \times 8 \times 256$.

The detailed breakdown of the parameters of the unwrapping model has been depicted in Table 11 in the Appendix, whilst the model architecture has been presented in Fig. 7. The amount of feature maps puts the number of total parameters of the model to $11,217,989 + 9 \times N_c$ split into ($11,211,477 + 9 \times N_c$ trainable + $6,512$ non-trainable parameters.)

4. Experiments

This section presents the result of experiments within the scope of the DL model training. First, the environments within which the models have been trained are described in Section 4.1. Sections 4.2 and 4.3 describe the process of generation and augmentation of the synthetic and real datasets respectively. Appendix C presents the metrics, with which the models have been evaluated and classified. Section 4.4 touches on training the U-Net model with 3 levels used for noise removal on synthetically generated data, whilst Section 4.5 describes training the semantic segmentation Attention U-Net model for PU by wrap count map generation.

4.1. Environment settings

The Attention U-Net model training has been performed on a stationary desktop PC. Its most crucial parameters for DL training are the GPU and CUDA software versions. The system used an NVIDIA GTX 1070 GPU with 8.0 GB of VRAM and CUDA software in version 11.4. The noise removal model has been developed and trained on the Google Colab platform using a high-end GPU dedicated to neural network computations, which has been made available by the platform.

4.2. Synthetic phase data

In the case of PU, the diversity of a synthetic dataset can be improved by implementing simulations of various aberrations and backgrounds. Images generated by the MATLAB script contain floating point data in the range $(0, 255]$. They were generated in three different ways ($I_{\text{spher}}(x, y)$, $I_{\text{lin + interp}}(x, y)$ and $I_{\text{spher + interp}}(x, y)$) and were further processed according to the data generation pipeline shown in Fig. 8.

$$I_{\text{spher}}(x, y) = W_x(x + W_{\text{posX}})^2 + W_y(y + W_{\text{posY}})^2 \tag{11}$$

$$I_{\text{lin + interp}}(x, y) = \alpha_{\text{BG}}(W_x(x + W_{\text{posX}}) + W_y(y + W_{\text{posY}})) + I_{\text{interp}} \tag{12}$$

$$I_{\text{spher + interp}}(x, y) = \alpha_{\text{BG}} I_{\text{spherical}}(x, y) + I_{\text{interp}} \tag{13}$$

where W_x and W_y have randomly initiated weights for x and y components, W_{posX} and W_{posY} are randomly generated positional coefficients and α_{BG} is a weight with which the interpolated image was combined with the generated backgrounds. $I_{\text{spher}}(x, y)$ denotes a spherical distribution, which was used in this dataset as a standalone phase image or as a background. I_{lin} is a linear component (a tilt or gradient). I_{interp} is an interpolated image, which was generated by randomly initializing a matrix of sizes between 3×3 and 11×11 and interpolating it to the size of 256×256 pixels, which was set as the image size in the training datasets, similarly to [27].

Rescaling the continuous phase provided a more accurate simulation of real data. This unwrapped phase was treated as the GT and served as the basis for the generation of: wrapped phase images (applying $mod2\pi$ on the unwrapped phase) and wrap count GT images (integer division by 2π). Noised images were created by adding noise sampled from a normal distribution

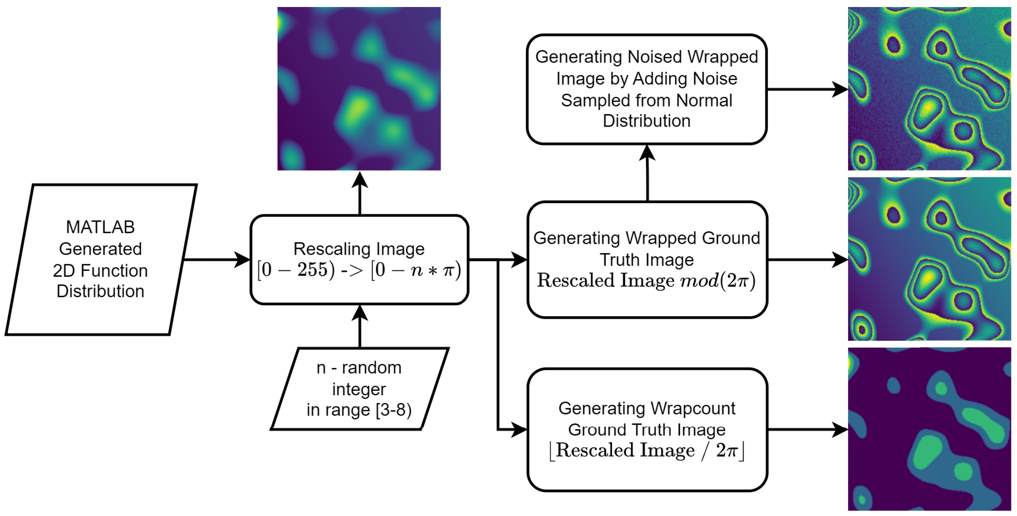
MATLAB Generated 2D Function Distribution
Rescaling Image $[0-255]$ -> $[0-n*\pi]$
n - random integer in range [3-8]
Generating Noised Wrapped Image by Adding Noise Sampled from Normal Distribution
Generating Wrapped Ground Truth Image Rescaled Image $mod(2\pi)$
Generating Wrapcount Ground Truth Image $\lfloor$Rescaled Image / $2\pi\rfloor$

Fig. 8. Synthetic data generation pipeline.

[56] $\mathcal{N}(\mu, \sigma^2)$ of which the probability density function $f_{\mathcal{N}}(x)$ is defined as in Eq. (14).

$$f_{\mathcal{N}}(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp[-\frac{1}{2}(\frac{x-\mu}{\sigma})^2] \tag{14}$$

where μ defines the mean of the distribution and σ denotes standard deviation. The mean of the noise was constant and set to 0.0, whilst the standard deviation was a randomly generated value in the range $[0.0, 0.6]$.

Generation of wrapped, wrap count, and noisy synthetic images have been performed in Python with the NumPy library. The different phase images generated this way have been showcased in Fig. 9.

4.3. Experimental phase data

Experimental phase data has been acquired in the form of digital holographic projections captured in the HT measurements at the Warsaw University of Technology [57]. The real dataset contains images of cells [58], organoids [37], phantoms [59,60] and regular structures that were 3D printed and exhibit high scattering properties [36]. The real dataset's continuous phase distribution has been obtained by PU with the QGPU algorithm [3]. In its entirety, the dataset contains 27,189 images. Each wrapped phase input image has its task-dependent counterpart (denoised, wrap count, unwrapped). The size of this dataset has been achieved by data augmentation using:

- rotation in 90° increments,
- rescaling (doubling and quadrupling in range),
- flipping around the vertical, horizontal, and diagonal axes,
- cropping from each corner and the middle of the image, each cropped image has been rescaled, flipped, and rotated.

The type and parameters of each operation have been assigned randomly. This way, one raw phase image could be used to generate, on average, more than 70 new dataset entries (Fig. 10).

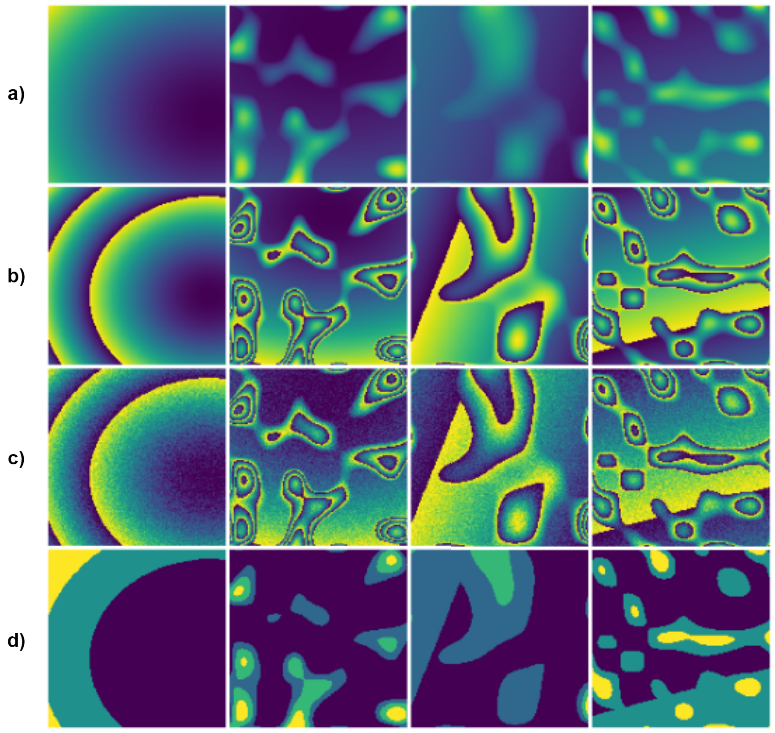

Fig. 9. Showcase of data from the synthetic dataset. Row a) represents rescaled continuous phase data, b) wrapped phase images, c) noised wrapped phase images, and d) shows wrap count maps for corresponding phase maps. Columns represent different phase maps in the same forms.

4.4. Experiments with noise removal

As mentioned in Section 4.1, training of the noise removal model has been performed using GPU cloud computing in the Google Colab platform. The dataset used for training this step was numerically generated, as described in Section 4.2. The dataset consists of 8,000 input-output image pairs split into training and validation sets in 80 : 20 proportion, typical for supervised learning. Wrapped phase image with synthetically generated noise was the input image, whilst the output was a noise-free phase image. The network's objective was noise removal through MSE error function minimization. Initial number of convolution filters on the shallowest depth level of the U-Net was defined as 8 and the size of the kernel in all convolution operations was 3×3. As described in Section 3.2, the number of feature maps generated through convolution doubled with each level of depth, whilst their dimensions were halved. This model also incorporated L2 kernel regularization. It introduces a penalty to the weight decay in the form of a squared value of the regularization coefficient.

Training was done with the Adam optimizer. Its name is an acronym for 'adaptive moment estimation'. Nowadays common in CNN training, Adam optimizer's advantage comes in memory requirements. It computes individual adaptive learning rates for estimates of gradient moments [61]. It was defined with two parameters: initial learning rate, which was set to 10^{-3}, and epsilon, a variable introduced for the sake of numerical stability, of value 10^{-7}. Epoch limit of the training was set to 200. During each epoch, batches of 16 images were introduced and afterwards, the weights were updated. Model was trained using the *keras* library, and it was monitored using three predefined callbacks. Each time a model's validation loss reached a new minimum, the

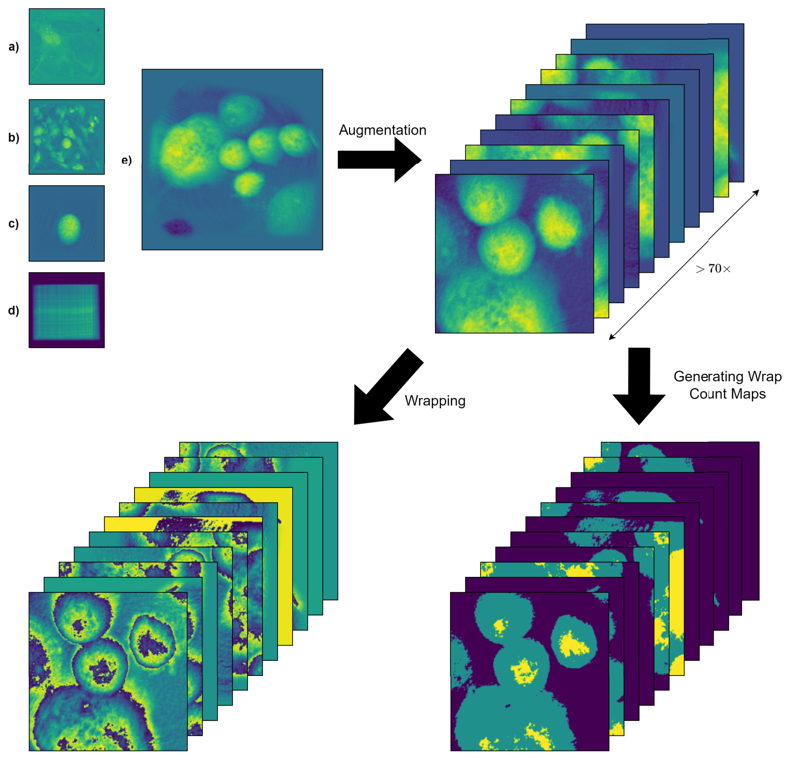

Fig. 10. Visualization of real dataset generation with example data from the experimental phase image dataset: a) cell #1, b) cell #2, c) spherical phantom, d) scattering structure [36], e) cell #3.

model was saved. If the validation loss has not improved for 15 epochs, the learning rate would decrease by a factor of 10, and if it did not improve over the last 20 epochs, learning stopped. The learning rate decreased at epochs 27, 49, and 60, reaching a final value of 10^{-6}. Final model was saved after the 62nd epoch, after which the validation loss (Eq. (6)) reached a value of 0.00432.

4.5. Experiments with wrap count map generation

Dataset used for training the unwrapping model consisted entirely of experimental phase images of living cells, organoids, and phantom structures acquired by HT, as described in Section 4.3. For this task wrapped phase image was the input data, and the GT for the segmentation task was the wrap count map. The whole dataset generated this way had 27,000 input-output image pairs. As in Section 4.4, the dataset was split into training and validation data by ratio $80 : 20$. Overfitting has been tackled by introducing an L1 regularization as a penalty to the kernel weights. Differing from the L2 regularization, L1 regularization introduces an absolute value of the regularization coefficient. This method has been applied to all the convolution kernels within the model, whose size has been set to 5×5. Similarly to the noise removal model, the initial number of kernels on the shallowest depth level has been defined as 8. The loss function used for training this model has been set to Sparse Categorical Cross-Entropy (SCE). It differs from Categorical Cross-Entropy (CE) in the output type. SCE takes integers as class labels, whilst in CE, the labels have to be one-hot-encoded. E.g., while choosing from 5 possible labels, 2*nd* label being correct: in SCE, that would be denoted by 1 (starting from 0), however, in CE it would be denoted as a vector $[0, 1, 0, 0, 0]$.

As in the denoising model, the optimizer and initial learning rate values were declared as Adam and 0.001, correspondingly. Epsilon value of 10^{-7} has also remained. Training has been limited to 300 epochs, during which batches of 32 data pairs have been introduced, however, because of the callback functions used for monitoring training, it ended after 93 epochs. Best results have been achieved after the 58^{th} epoch. At this point, the validation loss achieved its minimum at 0.13226 and accuracy at 96.448%. Throughout the training, the learning rate changed, according to a predefined callback, by a factor of 0.1. Decreasing at epochs 58, 76, and 93, finally reaching a value of 10^{-6}. Training time on a local desktop PC (section 4.1) was 4 hours and 5 minutes.

5. Results and discussion

The proposed pipeline has been tested on images from two origins. Section 5.1 describes the pipeline's performance in noise removal and unwrapping on synthetically generated data. Results on experimental phase images have been shown in section 5.2.

5.1. Results on synthetic data

This section starts with the depiction of the noise removal step of the proposed pipeline, which is followed by a presentation of the results of unwrapping on different phase distributions. Results of the models' denoising performance have been obtained from the same simulated phase distribution with added noise. Figure 11 displays a chosen phase image with 3 levels of noise.

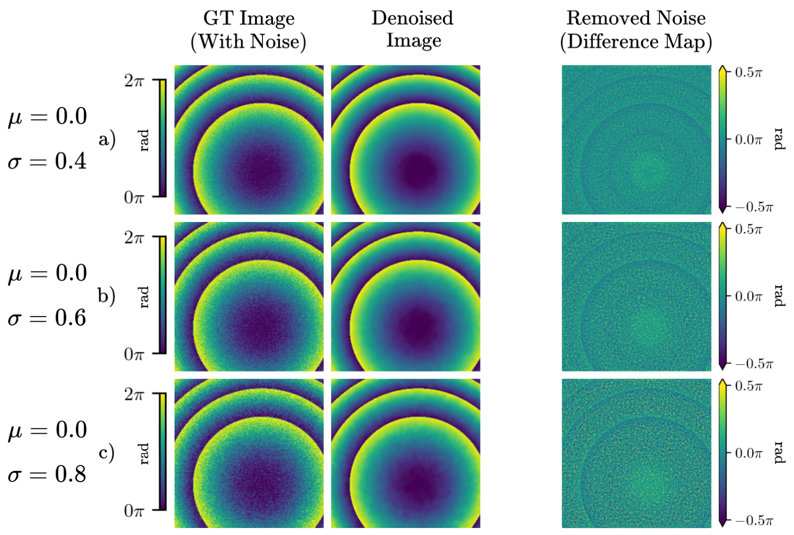

Fig. 11. Showcase of the noise removal step on 3 different levels of noise (varying in the standard deviation of its distribution): a) $\mu = 0.0, \sigma = 0.4$, b) $\mu = 0.0, \sigma = 0.6$ and c) $\mu = 0.0, \sigma = 0.8$. The difference map shows the amount of noise removed by the proposed model described in Section 3.2.

Figure 11 shows the performance of the proposed noise removal model on 3 phase images with different levels of noise, which has been simulated randomly within a normal distribution. Within each row (a - c), the mean μ has remained constant, whilst the standard deviation σ has increased. It should be noted that the denoising process removes noise, whilst preserving high-frequency information at the discontinuities. This feature is clearly shown in Figs. 18 and 19 in Appendix B.This figure shows a vertical cross-section taken through the center of the phase images.

Table 2 shows the metric values of denoising images shown in Fig. 11. Since this section reviews the results of the simulated phase images, the GT image (without any added noise beforehand) was available. Hence the $PSNR_{\text{noisy}}$ metric was computable. $PSNR_{\text{denoised}}$ compares the denoised images' quality with respect to the GT. Other metrics include MSE, RMSE, and SSIM. The metrics table shows that, independently of noise level, the model provides equal performance (improvement of $6 - 8$ dB of PSNR), which facilitates the following step of PU.

Table 2. Metric values for noise removal of images with different noise levels (Fig. 11(a), (b), (c)).

	$PSNR_{\text{noisy}}$	$PSNR_{\text{denoised}}$	MSE [rad^2]	RMSE [rad]	SSIM
a) ($\mu = 0.0, \sigma = 0.4$)	20.0818 dB	27.3025 dB	0.0075	0.0863	0.9559
b) ($\mu = 0.0, \sigma = 0.6$)	18.4110 dB	26.8887 dB	0.0085	0.0923	0.9539
c) ($\mu = 0.0, \sigma = 0.8$)	17.2508 dB	26.0882 dB	0.0115	0.1073	0.9319

Figure 12 presents the results of unwrapping of 3 chosen simulated phase images with the proposed method: a) spherical phase distribution resulting in closed phase fringes on the wrapped image, b) phase distribution resembling cells and a background (BG) with a spherical phase component, and c) phase distribution resembling cells and a BG with a slight phase tilt. First and 4th column shows the GT images obtained during dataset generation, whilst the 2nd and 3rd columns depict the model's prediction and unwrapped image correspondingly. The 5th column shows an unwrapping error map between the unwrapped and GT images.

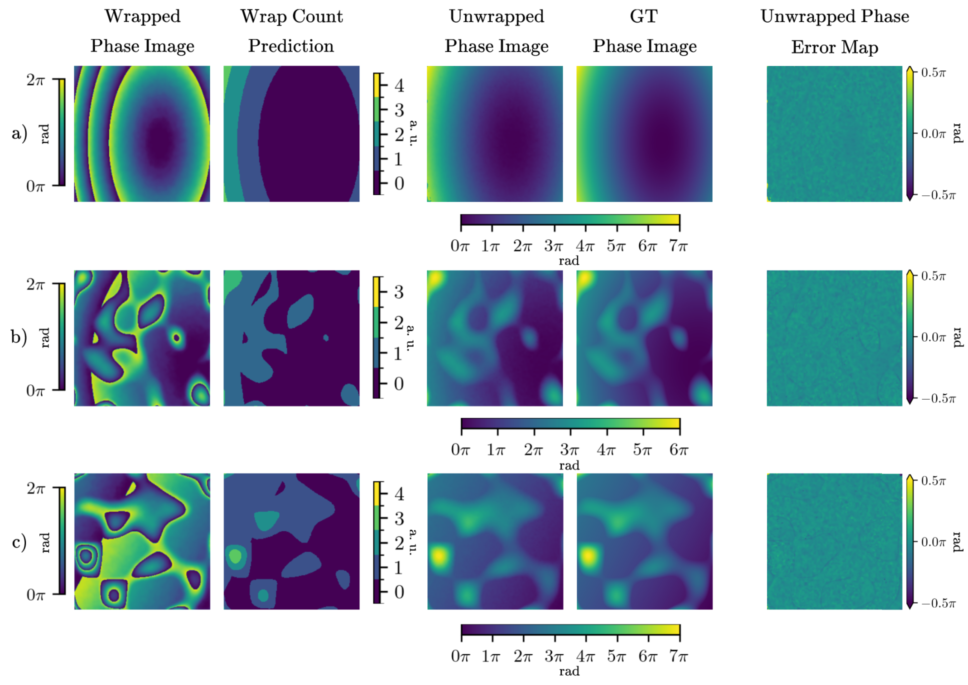

Fig. 12. Showcase of PU by predicting wrap count map through semantic segmentation with the proposed model described in Section 3.3.

The error map, especially in rows b) and c), shows that the phase is unwrapped with good performance even with sudden and steeper gradients of phase (e.g., simulated cell over the background with linear gradient). This can also be observed on the left-hand side of the unwrapped image in row c), where the phase gradient is steep, and the phase fringes on the corresponding wrapped phase image are dense.

Results presented in this section show phase images that have been denoised and unwrapped. This demonstrates the performance of the entire proposed pipeline. However, it also carries several consequences in evaluating the pipeline's performance. Since the noise removal step has affected the unwrapped and denoised image compared to a GT continuous phase image, metrics like the MSE, RMSE, and SSIM are visibly affected. The nature of unwrapping through segmentation of wrap count maps means that (using only the unwrapping step) the errors stem from pixel misclassifications, and thus the errors on the unwrapped phase can only be equal to multiples of 2π. Denoising of images before unwrapping them causes the denoising error to be propagated, and its effects are visible on the metrics of evaluation. However, since the objective of this section is to evaluate the entire pipeline, the authors decided against changing the evaluation but adding an explanation.

Testing on synthetic phase images enabled comparison of PSNR of the final step (denoised and unwrapped phase image compared to the GT continuous phase image). This metric was described in the final column of Table 3. Other metrics include the MSE, RMSE, SSIM, and the segmentation accuracy for the GT segmentation map. The table shows that the model provides exceptional results on the tested images. In fact, very few pixels were misclassified. $PSNR_{\mathrm{unwr}}$ proves that removing the images' noise constitutes to obtaining a good result. Other metrics also show that the unwrapped images very closely resemble the GT image; however, bigger values of the MSE and RMSE or smaller values of the SSIM show that, as mentioned before, the error is being propagated from the noise removal step.

Table 3. Metrics values for PU for 3 different types of simulated images (Fig. 12(a), (b), (c)).

	MSE [rad^2]	RMSE [rad]	SSIM	ACC_{seg}	$PSNR_{\mathrm{unwr}}$
Spherical distribution **a)**	0.0423	0.2057	0.9879	99.9069%	33.6324 dB
Cells w/ spherical BG **b)**	0.0116	0.1079	0.9846	99.9924%	35.1077 dB
Cells w/ tilted BG **c)**	0.0301	0.1735	0.9905	99.9481%	34.3571 dB

5.2. Results on experimental data

Several types of experimental phase images have been chosen in this section. This section has been split into showcasing the performance of both steps of the proposed pipeline. Firstly, the authors focus on noise removal on real data, which is followed by a display of results of the second step of the pipeline - unwrapping by semantic segmentation.

Figure 13(a), (b), and (c) depict different types of experimental phase fringe images used for testing the pipeline, more importantly, its' first step. The first row (phase image a)) depicts challenging data - a high scattering foam 3D printed using NanoScribe [57], 2nd and 3rd rows b and c show phase images of two different cells, acquired with an HT technique. The images are denoised by the model trained on the synthetic dataset, which is discussed in sections 3.2 and 4.4.

The experimental phase images that were taken using the HT technique exhibit high irregularities due to the measured specimens' natural irregularity in structure. This means that unwrapping a realistic wrapped phase image is significantly more difficult than doing so on simulated phase distributions. This is visible especially in row a) of Fig. 13, where the specimen was a highly scattering 3D printed object. Its irregular surface resulted in strongly irregular discontinuities.

However, even in these extreme cases, the model is capable of preserving phase information at the discontinuities, while smoothing out and removing noise within the structure. It is clearly illustrated in Fig. 19 (cross-sections obtained by slicing vertically through the images' center) in Appendix B.

Metrics marked with * can also be interpreted differently. In the simulated case, denoised phase images are directly compared to their simulated noise-free counterpart. Real data analysis lacks the noise-free GT image, which means that the metrics shown in Table 4 reflect a comparison

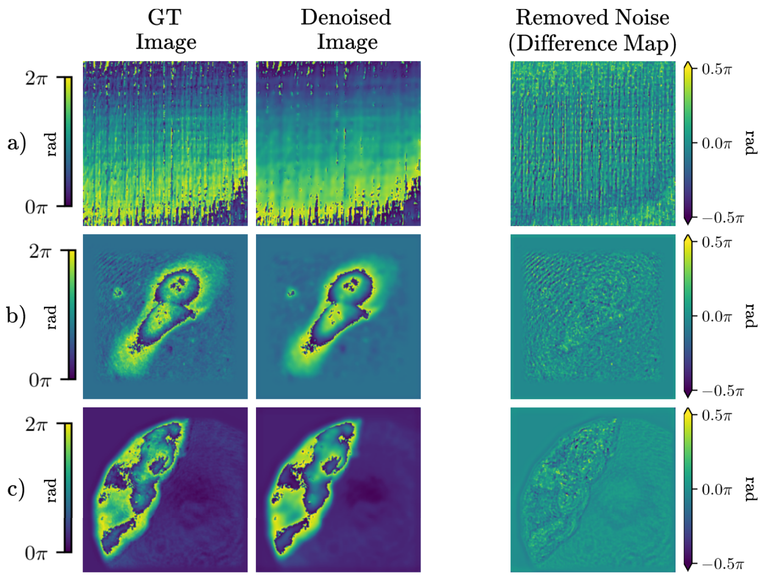

Fig. 13. Showcase of the noise removal step on different types of real data, which was described in Section 4.3. The difference map shows the amount of noise removed by the proposed model described in Section 3.2; a) scattering 3D printed object, b) cell #1, c) cell #2.

between phase images with noise and those processed by the proposed denoising model. This is also why the $PSNR_{\text{noisy}}$ is not computable. Phase images that originate from an HT measurement of the high-scattering 3D printed foam proved to be the most difficult to correctly process, which is confirmed by the metrics values (MSE and RMSE values are the biggest, whilst the SSIM is below 0.5). Rows b) and c) show that noise removal does not impact the visibility of the structure of the measured specimen; the details are still visible.

Table 4. Metric values for noise removal of 3 experimental phase images (Fig. 13(a), (b), (c)).

	$PSNR^*_{\text{noisy}}$	$PSNR^*_{\text{denoised}}$	MSE* [rad^2]	RMSE* [rad]	SSIM*
Scattering 3D printed object **a)**	– dB	19.6442 dB	0.3347	0.5786	0.4445
Cell #1 **b)**	– dB	22.8006 dB	0.0877	0.2961	0.5208
Cell #2 **c)**	– dB	23.6758 dB	0.0533	0.2309	0.6579

Unwrapping is shown on 4 phase images. Figure 14 shows a wrapped phase image of a spherical phantom (a) and cell images acquired using HT (b); Fig. 15 shows a cropped phase image of a 3D printed foam exhibiting high scattering level (a) (similarly to Fig. 13(a)) and a phase image of nasal epithelium cell (b).

The metrics used to evaluate unwrapping on experimental phase images are the same as in Section 5.1. It can be seen that the segmentation part is performed with high accuracy for rows a) and b) in Fig. 14, for which the proper GT wrap count map was available. Due to the direct comparison with the GT unwrapped phase, other metrics are decreased, resulting from the direct error propagation from the denoising step (described in detail in Section 5.1). It can be seen that these metrics (MSE, RMSE, and SSIM) values get worse lost during processing through the pipeline. However, this quality loss only corresponds to noise removal (the GT image is an unwrapped image on which no noise removal has been performed). Table 5 shows that the model used for segmentation is capable of a perfect prediction on an experimental phase image.

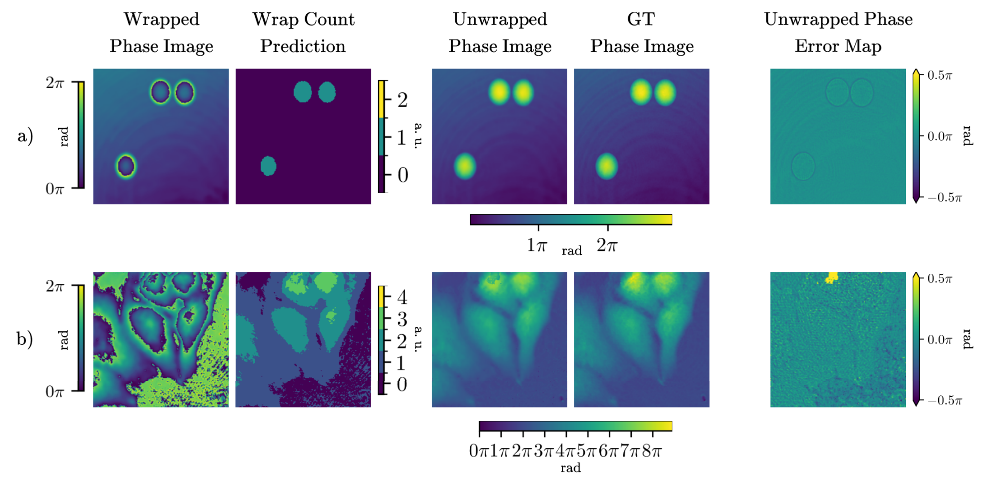

Fig. 14. Showcase of the result of the PU step on different types of real data, which was described in Section 4.3. The error map shows the amount of noise removed by the proposed model described in Section 3.2 combined with unwrapping error induced by the model described in Section 3.3.

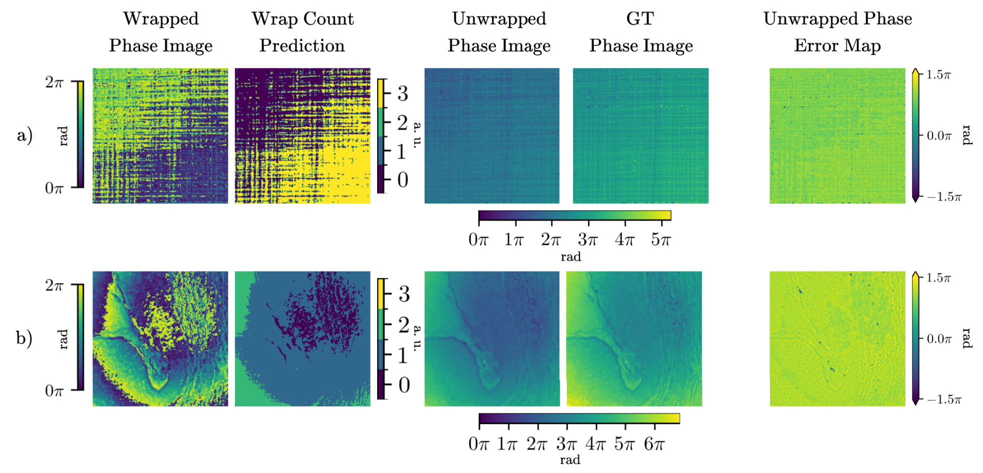

Fig. 15. Showcase of performance of the pipeline tested on data obtained similarly to that on Fig. 14. Errors on the error maps originate from the poor performance of the benchmark algorithm, which is used to generate GT phase images.

Table 5. Metric values for PU for two different types of experimental phase images (Fig. 14(a), (b)).

	MSE [rad^2]	RMSE [rad]	SSIM	ACC_{seg}	$PSNR_{\text{unwr}}$
Spherical phantom **a)**	0.0044	0.0669	0.9733	100%	30.7664 dB
Cell #3 **b)**	0.3398	0.5829	0.9281	99.0356%	30.3202 dB

Evaluation of experimental phase images can be difficult, especially whilst examining complicated phase distributions. Although required, obtaining a GT image for comparison with the prediction of the proposed solution generates problems with complex and irregular phase images. Figure 15 and Table 6 demonstrate this phenomenon. The GT image is taken by unwrapping the phase using a QGPU [3] algorithm, which is treated as a baseline method for comparison in this paper. However, this method is not perfect, and highly irregular phase fringes (Fig. 15(a) and (b)) can surpass its capabilities. These wrapped phase images have not been properly unwrapped by the baseline. Hence the metrics in Table 6 are not representative.

Table 6. Metric values for PU for two different types of experimental phase images (Fig. 15(a), (b)).

	MSE [rad^2]	RMSE [rad]	SSIM	ACC_{seg}	$PSNR_{\text{unwr}}$
Scattering 3D printed object **a)**	9.1220**	3.0202**	0.7168**	50.5829%**	17.1177 dB**
Nasal epithelium **b)**	12.9663**	3.6009**	0.7893**	99.2737%**	19.3662 dB**

Metrics values of results from the unwrapping of phase images, marked with **, on Fig. 15(a) and (b) (Table 6) are compromised due to not obtaining proper GT images for comparison. Wrapped phase distribution was complex enough to generate issues for the QGPU algorithm [3], used for generating all experimental phase GT images.

5.3. Ablation study

In order to verify the importance of the proposed elements within our model, we performed an ablation test, during which three new models were assessed: a model without AGs, a model without RBs, and a model without both AGs and RBs. What is important, the training parameters have been consistent with the original model for all 3 new models within the ablation study. The dataset has also been split (80:20, as mentioned in section 4.5) in the same way (the parameter *random_state* has been preserved).

This study has been performed on a testing dataset of 100 images, from which the *mean* μ and *standard deviation* σ of the metric values have been chosen for analysis. The metrics discussed in this study are *ACC* (Eq. (C5)) and *MSE* (Eq. (6)). The results have been shown in Table 7.

Table 7. MSE (Eq. (6)) and ACC (Eq. (C5)) statistical values for PU of the ablation testing dataset performed with different models.

	Full model	Model w/o RBs	Model w/o AGs	Model w/o RBs & AGs
MSE [rad^2]	0.0512 ± 0.0754	1.7728 ± 2.8042	0.0421 ± 0.0531	1.9254 ± 3.2574
ACC [%]	99.9055 ± 0.2882	96.0651 ± 6.0273	99.9257 ± 0.2592	95.3864 ± 7.8052

It can be deduced from Table 7 that the implementation of AGs into our unwrapping U-Net model did not have a significant influence. Overall performance of the complete model and the one without AGs remains on a similar level and it does so with similar consistency. Note that the ablation testing dataset consisted of both synthetic and experimental phase images. This helps draw an important conclusion that AG is not necessary for PU of HT images.

At the same time, this table shows the importance of RBs in PU of HT data. The model without RBs and the model without both RBs and AGs provide subpar results on this dataset. The mean values of the metrics show the models poor performance and inability to generalise on new data, whilst the standard deviation show their inconsistency throughout the study.

5.4. Comparison against state-of-the-art methods

Figure 16 shows the results of a direct comparison of segmentation wrap count maps between State-of-the-Art models. As indicated in Section 2.2.2, we compare the DL-based solutions, which rely on the semantic segmentation task. The chosen models are: (1) the U-Net model proposed in [28], which is the original U-Net model proposed for medical image segmentation and which is the base for our proposed solution; (2) PhaseNet [41], which also uses the wrap count map segmentation for PU; (3) PhaseNet2.0 [26], which is a follow-up work to PhaseNet; (4) the method denoted as PU-DCSN, which was proposed in [43] and its name is the abbreviated form of "Phase Unwrapping - Denoised Convolutional Segmentation Network" for ease of reference; (5) model for simultaneous denoising and wrap count map generation named TriNet [62] by Sumanth et al.; (6) EESANet [63] developed for phase unwrapping and also using the self-attention mechanism. The PU-DCSN method uses a similarly structured pipeline consisting of noise removal and wrap count map generation through semantic segmentation.

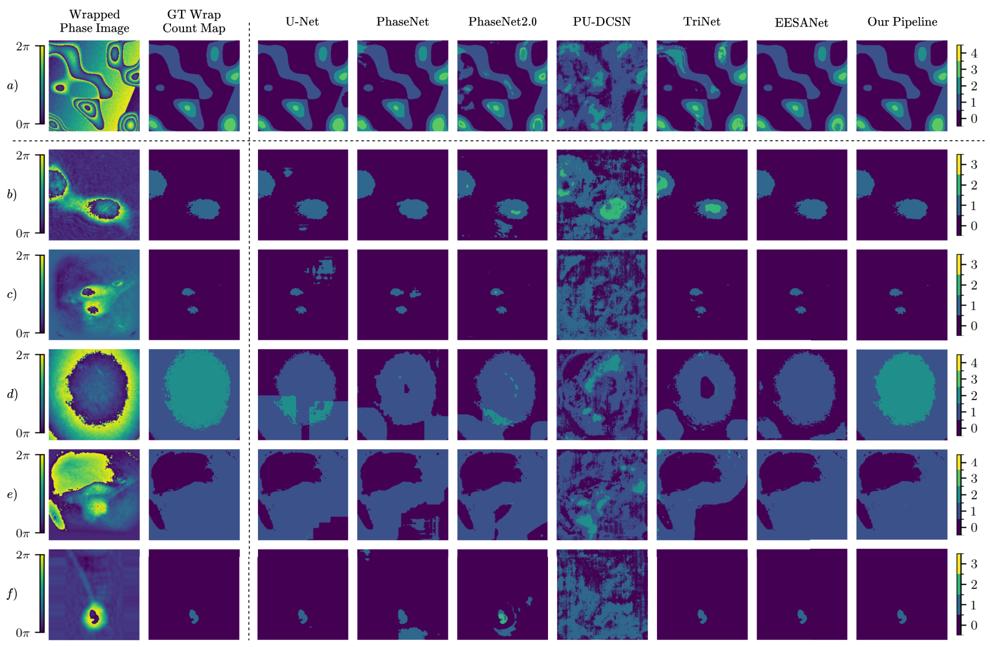

Fig. 16. Comparison with other methods of PU (semantic segmentation of wrap count maps).

Note that the PhaseNet, PhaseNet 2.0, PU-DCSN, TriNet, and EESANet models were implemented to the best of the abilities of the authors. The implementation is based solely on the models' descriptions in their corresponding papers, which is caused by code unavailability. The models were successfully coded in Keras (Python), compiled, trained, and used to predict the wrapped phase data. It should also be noted, that no preprocessing step has been introduced to solutions that did not include it. This is based on the idea, that the authors compare the entire solution for DL-based PU for HT phase images, rather than just the unwrapping segmentation model.

Figure 16 shows that whilst the U-Net (3rd column) is capable of correct predictions on this dataset, its results were not consistent and performance was poorer than that of our pipeline. Similarly, PhaseNet and PhaseNet 2.0 (4th and 5th columns, respectively) predicted the wrap count maps roughly correctly. However, the results contained misclassified wrapped areas, where the phase distribution exhibited no discontinuities. PU-DCSN (6th column) has not shown proper segmentation on any wrapped phase image. Such strong errors indicate that the method has not been reproduced successfully. The 7th and 8th columns present results obtained by the TriNet and EESANet models correspondingly. Results obtained by implementing our proposed pipeline have been placed in the 9th column. The figure indicates that our pipeline can produce the best results on the irregular phase distributions of different types and magnitudes. This conclusion is also backed in Table 8, where each of the predicted wrap count maps presented in Fig. 16 is compared to the GT wrap count map of the corresponding wrapped phase image. The metric used for this table is the prediction accuracy (Eq. (C5)). It can be seen that, according to the accuracy metric, the proposed pipeline performs with the most consistency and is on par with the EESANet model in terms of high quality wrap count map generation.

Table 8. Comparison between different semantic segmentation DL methods used in literature for PU. The metric used for this comparison is the prediction accuracy (Eq. (C5)). The best metric between DL-based solutions is emphasized in bold. The rows a-f refer to the rows a-f on Fig. 16.

	U-Net [28]	PhaseNet [41]	PhaseNet2.0 [26]	PU-DCSN [43]	TriNet [62]	EESANet [63]	Our Pipeline
a)	98.9944%	95.5978%	90.8585%	39.6469%	92.4973%	99.3805%	**99.9527%**
b)	98.9700%	99.3439%	97.4823%	44.1315%	97.1756%	**100%**	**100%**
c)	96.5759%	99.1013%	99.9054%	41.3849%	99.8932%	**100%**	**100%**
d)	35.9924%	17.8970%	18.2922%	35.5896%	13.8657%	9.0942%	**92.1082%**
e)	91.5436%	81.3187%	94.3298%	58.1177%	73.5962%	**100%**	99.9985%
f)	99.9420%	95.6604%	97.0856%	44.1498%	**100%**	**100%**	**100%**

The final step of PU through semantic segmentation is multiplying the predicted wrap count map by 2π and adding it to the wrapped phase map. Figure 17 shows the unwrapped phase images of each method mentioned in this section. Their notation and placement correspond to those in Fig. 16. Additionally, two conventional methods have been shown for comparison. These are QGPU [3], used as a baseline for obtaining experimental phase GT images, and MST [4]. Both have been described in depth in Section 2.2.1. Table 9 reinforces the observations stated in this section by comparing the unwrapped phase images' MSE value concerning their GT continuous phase counterparts. It can be seen that the smallest MSE value has been achieved by the EESANet model. However, the proposed pipeline provides good results more consistently throughout the entire set of testing images. The larger MSE values originate from the denoising operation and are propagated through to the final result (as discussed in section 5.1).

Table 9. Comparison between different PU methods (conventional and DL-based). The metric used for this comparison is the MSE [rad^2] (Eq. (6)). The best metric between DL-based solutions is emphasized in bold. The rows a-f refer to the rows a-f on Fig. 17.

	U-Net [28]	PhaseNet [41]	PhaseNet2.0 [26]	PU-DCSN [43]	TriNet [62]	EESANet [63]	Our Pipeline
a)	0.5423	1.8976	3.7781	27.7800	2.8421	0.3855	**0.0301**
b)	0.4066	0.2590	0.9939	22.8907	1.0868	**$3.3232*10^{-15}$**	0.0145
c)	1.3518	0.3548	0.0373	23.3286	0.0989	**$4.6741*10^{-16}$**	0.0084
d)	27.7232	38.0909	36.3195	38.7747	44.9349	40.8727	**0.3685**
e)	3.3439	7.3751	2.2385	16.7861	10.7847	**$2.2534*10^{-14}$**	0.0092
f)	0.0229	1.7132	1.1506	22.0080	0.0153	**$3.1114*10^{-16}$**	0.0006

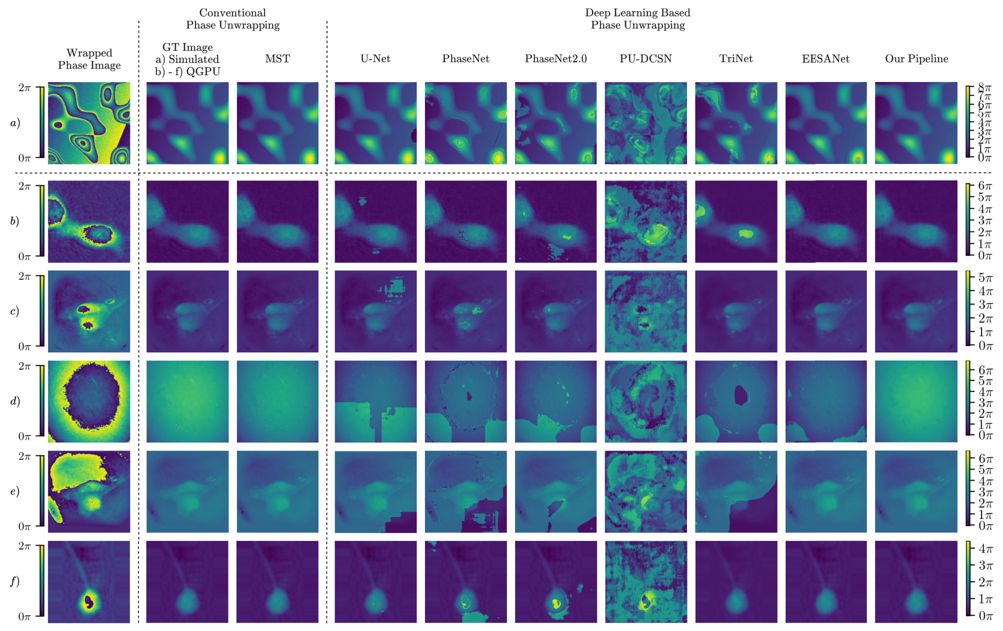
Conventional Phase Unwrapping
Deep Learning Based Phase Unwrapping
Wrapped Phase Image
GT Image a) Simulated b) - f) QGPU
MST
U-Net
PhaseNet
PhaseNet2.0
PU-DCSN
TriNet
EESANet
Our Pipeline
a)
b)
c)
d)
e)
f)
2π
0π

Fig. 17. Comparison with other methods of PU.

6. Conclusions

In this paper, we propose the PU pipeline consisting of a noise removal step and a PU step. The PU step is done by semantic segmentation of areas limited by phase discontinuities, described as phase wrap counts. The implementation of AGs and RBs in the model has improved the unwrapping capabilities of the model. In addition, we describe the process of obtaining, generating, and augmenting synthetic and real datasets used for training and evaluating the models. The method is tailor-made for HT applications and has shown great results and promise for analysing phase images of living cell specimens, phantoms, and complex structures exhibiting high degree of scattering. It should be emphasised that this article not yet provides the general rules regarding the DL-based denoising and PU.

The results are compared with four State-of-the-Art DL methods based on the semantic segmentation technique. This comparison shows that our proposed pipeline exhibits the best PU and denoising capabilities for highly irregular data, which come from different HT measurements of various specimens.

In the future, this new unwrapping approach will be fully implemented in the HT system with the aim to obtain improved 3D RI reconstructions for biological microobjects (cell monolayers, organoids, tissues) with high scattering features.

Appendix A

This appendix contains tables, which describe the parameters of the denoising and unwrapping DL models described in this paper in sections 3.2 and 3.3, respectively.

Table 10. Breakdown of parameters of the model used for noise removal (Fig. 5). Kernel size has been defined as 5×5, containing 25 parameters. Abbreviations used in the table: CB - 2D Convolutional Block (Fig. 4(a)), RB - Residual Block (Fig. 4(b)), TC - 2D Transposed Convolution, FinCB - Final 2D Convolutional Layer, b - biases.

Depth Level	Encoder		Decoder	
	Size Per Layer	**Total Per Level**	**Size Per Layer**	**Total Per Level**
1	**CB:** $1 \times 8 \times 25 + 8 \times 4$ **RB:** $2 \times (8 \times 8 \times 25 + 8 \times 4)$ **CB:** $8 \times 8 \times 25 + 8 \times 4$ **b:** 4×8	5160	**TC:** $16 \times 8 \times 25$ **CB:** $16 \times 8 \times 25 + 8 \times 4$ **RB:** $2 \times (8 \times 8 \times 25 + 8 \times 4)$ **CB:** $8 \times 8 \times 25 + 8 \times 4$ **FinCB:** $8 \times 1 \times 1$ **b:** $5 \times 8 + 1$	11377
2	**CB:** $8 \times 16 \times 25 + 16 \times 4$ **RB:** $2 \times (16 \times 16 \times 25 + 16 \times 4)$ **CB:** $16 \times 16 \times 25 + 16 \times 4$ **b:** 4×16	22720	**TC:** $32 \times 16 \times 25$ **CB:** $32 \times 16 \times 25 + 16 \times 4$ **RB:** $2 \times (16 \times 16 \times 25 + 16 \times 4)$ **CB:** $16 \times 16 \times 25 + 16 \times 4$ **b:** 5×16	45136
3	**CB:** $16 \times 32 \times 25 + 32 \times 4$ **RB:** $2 \times (32 \times 32 \times 25 + 32 \times 4)$ **CB:** $32 \times 32 \times 25 + 32 \times 4$ **b:** 4×32	90240	**TC:** $64 \times 32 \times 25$ **CB:** $64 \times 32 \times 25 + 32 \times 4$ **RB:** $2 \times (32 \times 32 \times 25 + 32 \times 4)$ **CB:** $32 \times 32 \times 25 + 32 \times 4$ **b:** 5×32	179872
	Bottleneck			
	Size Per Layer		**Total Per Level**	
4	**CB:** $32 \times 64 \times 25 + 64 \times 4$ **RB:** $2 \times (64 \times 64 \times 25 + 64 \times 4)$ **CB:** $64 \times 64 \times 25 + 64 \times 4$ **b:** 4×64		359680	
	Total Number of Parameters			714,185 (712,777 trainable + 1,408 non-trainable)

Table 11. Breakdown of parameters of the Attention U-Net model used for wrap count map segmentation (Fig. 7). Kernel size has been defined as 5×5, containing 25 parameters, except for the 2D Transposed Convolutions, which used 3×3 sized kernels, containing 9 parameters. Abbreviations used in the table: CB - 2D Convolutional Block (Fig. 4(a)), RB - Residual Block (Fig. 4(b)), TC - 2D Transposed Convolution, AG - Attention Gate (Fig. 6), FinCB - Final 2D Convolutional Layer, b - biases, N_c - number of classes defined in the segmentation task.

Depth Level	Encoder		Decoder	
	Size Per Layer	**Total Per Level**	**Size Per Layer**	**Total Per Level**
1	**CB:** $1 \times 8 \times 25 + 8 \times 4$ **RB:** $2 \times (8 \times 8 \times 25 + 8 \times 4)$ **CB:** $8 \times 8 \times 25 + 8 \times 4$ **b:** 4×8	5160	**TC:** $16 \times 8 \times 9$ **CB:** $16 \times 8 \times 25 + 8 \times 4$ **RB:** $2 \times (8 \times 8 \times 25 + 8 \times 4)$ **CB:** $8 \times 8 \times 25 + 8 \times 4$ **AG:** $16 \times 8 \times 9+$ $2 \times (8 \times 8 \times 1)+$ $8 \times 1 \times 1 + 8 \times 4$ **FinCB:** $8 \times N_c \times 1$ **b:** $8 \times 8 + 1 + N_c$	$10620 + 9 \times N_c$
2	**CB:** $8 \times 16 \times 25 + 16 \times 4$ **RB:** $2 \times (16 \times 16 \times 25 + 16 \times 4)$ **CB:** $16 \times 16 \times 25 + 16 \times 4$ **b:** 4×16	22720	**TC:** $32 \times 16 \times 9$ **CB:** $32 \times 16 \times 25 + 16 \times 4$ **RB:** $2 \times (16 \times 16 \times 25 + 16 \times 4)$ **CB:** $16 \times 16 \times 25 + 16 \times 4$ **AG:** $32 \times 16 \times 9+$ $2 \times (16 \times 16 \times 1)+$ $16 \times 1 \times 1 + 16 \times 4$ **b:** $8 \times 16 + 1$	42193
3	**CB:** $16 \times 32 \times 25 + 32 \times 4$ **RB:** $2 \times (32 \times 32 \times 25 + 32 \times 4)$ **CB:** $32 \times 32 \times 25 + 32 \times 4$ **b:** 4×32	90240	**TC:** $64 \times 32 \times 9$ **CB:** $64 \times 32 \times 25 + 32 \times 4$ **RB:** $2 \times (32 \times 32 \times 25 + 32 \times 4)$ **CB:** $32 \times 32 \times 25 + 32 \times 4$ **AG:** $64 \times 32 \times 9+$ $2 \times (32 \times 32 \times 1)+$ $32 \times 1 \times 1 + 32 \times 4$ **b:** $8 \times 32 + 1$	167841
4	**CB:** $32 \times 64 \times 25 + 64 \times 4$ **RB:** $2 \times (64 \times 64 \times 25 + 64 \times 4)$ **CB:** $64 \times 64 \times 25 + 64 \times 4$ **b:** 4×64	359680	**TC:** $128 \times 64 \times 9$ **CB:** $128 \times 64 \times 25 + 64 \times 4$ **RB:** $2 \times (64 \times 64 \times 25 + 64 \times 4)$ **CB:** $64 \times 64 \times 25 + 64 \times 4$ **AG:** $128 \times 64 \times 9+$ $2 \times (64 \times 64 \times 1)+$ $64 \times 1 \times 1 + 64 \times 4$ **b:** $8 \times 64 + 1$	669505
5	**CB:** $64 \times 128 \times 25 + 128 \times 4$ **RB:** $2 \times (128 \times 128 \times 25 + 128 \times 4)$ **CB:** $128 \times 128 \times 25 + 128 \times 4$ **b:** 4×128	1436160	**TC:** $256 \times 128 \times 9$ **CB:** $256 \times 128 \times 25 + 128 \times 4$ **RB:** $2 \times (128 \times 128 \times 25 + 128 \times 4)$ **CB:** $128 \times 128 \times 25 + 128 \times 4$ **AG:** $256 \times 128 \times 9+$ $2 \times (128 \times 128 \times 1)+$ $128 \times 1 \times 1 + 128 \times 4$ **b:** $8 \times 128 + 1$	2674305
	Bottleneck			
	Size Per Layer		**Total Per Level**	
6	**CB:** $128 \times 256 \times 25 + 256 \times 4$ **RB:** $2 \times (256 \times 256 \times 25 + 256 \times 4)$ **CB:** $256 \times 256 \times 25 + 256 \times 4$ **b:** 4×256		5739520	
	Total Number of Parameters			$11{,}217{,}989 + 9 \times N_c$ $9 \times N_c$ trainable + 6,512 non-trainable)

Appendix B

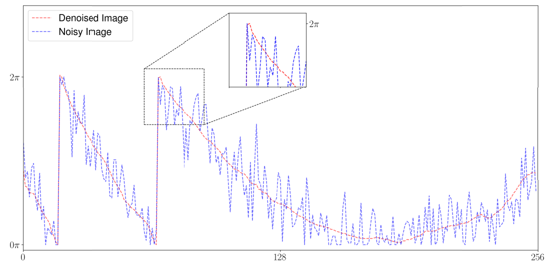

Fig. 18. Vertical cross-section taken through the center of a synthetic phase image in depicted in Fig. 11(c) ($\mu = 0.0, \sigma = 0.8$). Plot depicts an enhanced view of the phase discontinuities to show the high frequency information preservation.

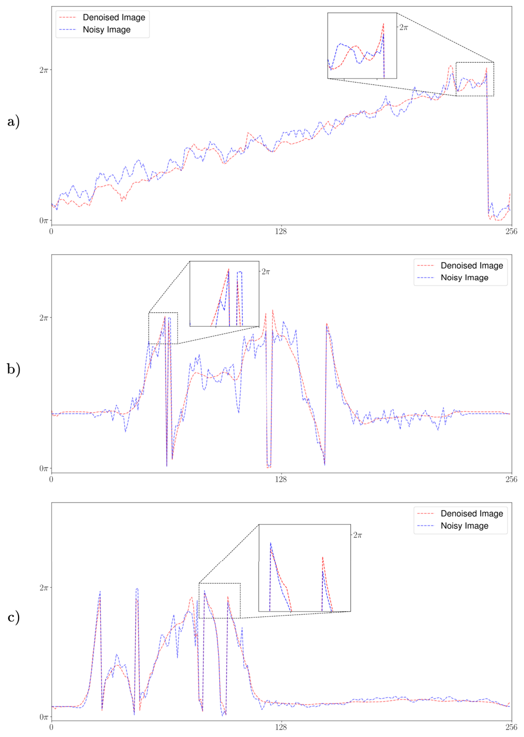

Fig. 19. Vertical cross-section taken through the center of a synthetic phase image in depicted in Fig. 13(a), b and c. Plot depicts an enhanced view of the phase discontinuities to show the high frequency information preservation.

Appendix C: performance metrics

In this article, 5 performance evaluation metrics, which were described in this section, were used. MSE (Eq. (6)) is often used in image translation tasks. It describes a pixel-wise mean squared difference between one matrix (network prediction) and a second matrix (GT image). It requires a continuous phase distribution for comparison. Root mean squared error (RMSE) image translation tasks. It is more easily interpretable, because it represents the magnitude of the response variable, as opposed to its squared value (MSE). Similar to MSE, the goal is minimisation of this metric. Since both methods are used for continuous distribution comparison, they have been chosen for this article.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=0}^{N}(t_i - s_i)^2}, \tag{C1}$$

where the notation is analogical to Eq. (6).

Structural Similarity Index (SSIM) has been developed by Wang et al. [65]. It describes the similarity between two images and is believed to be correlated with the perception quality of the human visual systems. Its values span the range (0, 1], with 0 representing no correlation between images. Hence the goal is to maximize SSIM value during training as well as the pipeline evaluation.

$$SSIM = l(f,g)c(f,g)s(f,g), \tag{C2}$$

where:

$$\begin{cases} l(f,g) = \frac{2\mu_f\mu_g + C_1}{\mu_f^2 + \mu_g^2 + C_1} \\ c(f,g) = \frac{2\sigma_f\sigma_g + C_2}{\sigma_f^2 + \sigma_g^2 + C_2} \\ s(f,g) = \frac{\sigma_{fg} + C_3}{\sigma_f\sigma_g + C_3} \end{cases} \tag{C3}$$

In Eq. (C2) the terms l, c and s represent comparisons of luminance, contrast, and structure of images f and g, as shown in Eq. (C3). μ and σ describe the mean and standard deviation of corresponding matrices. The variables C_1, C_2 and C_3 have been introduced to ensure a non-zero denominator [65,66].

Quality of denoising is done with a Peak Signal-to-Noise Ratio (PSNR) [66]. Increase of this metric describes a higher image quality. Since it requires a comparison with GT, it was only computed during denoising synthetic images.

$$PSNR = 10\log_{10}\left(\frac{{I_{\mathrm{range}}}^2}{\frac{1}{N}\sum_{i=0}^{N}(t_i - s_i)^2}\right), \tag{C4}$$

where variables are denoted analogically to Eqs. (6) and (C1).

Final metric used in this paper is classification accuracy, denoted as ACC. It describes the percentage of correctly predicted pixels with respect to the whole matrix.

$$ACC = \frac{N_c}{N_t}, \tag{C5}$$

where N_c and N_t denote the number of correctly classified pixels and total pixels correspondingly.

The models in this pipeline were evaluated using the beforementioned metrics. More specifically, the denoising model has been evaluated using MSE (Eq. (6)), RMSE (Eq. (C1)), SSIM (Eq. (C2)) and PSNR (Eq. (C4)) metrics. The unwrapping model's segmentation performance has been evaluated using the accuracy metric (Eq. (C5)), whilst the MSE (Eq. (6)), RMSE (Eq. (C1)) and SSIM (Eq. (C2)) have been used for the final unwrapping evaluation.

Funding. H2020 Industrial Leadership (101016726); Ministerstwo Edukacji i Nauki (PM/SP/0079/2021/1); Narodowa Agencja Wymiany Akademickiej (POWR.03.03.00-00-PN13/18).

Disclosures. The authors declare no conflicts of interest.

CRediT statement. **MG**: Conceptualization, Data Curation, Investigation, Methodology, Software, Validation, Visualization, Writing - Original Draft. **VD**: Formal Analysis, Supervision, Writing - Review & Editing. **MK**: Conceptualization, Funding acquisition, Project administration, Supervision, Writing - Review & Editing. **WK**: Conceptualization, Formal analysis, Project administration, Supervision, Writing - Review & Editing.

Data availability. Data underlying the results presented in this paper are available in [64].

References

1. G. Popescu, *Quantitative phase imaging of cells and tissues* (McGraw-Hill Education, 2011).
2. Y. Park, C. Depeursinge, and G. Popescu, "Quantitative phase imaging in biomedicine," Nat. Photonics **12**(10), 578–589 (2018).
3. M. Arevalillo-Herráez, F. R. Villatoro, and M. A. Gdeisat, "A robust and simple measure for quality-guided 2d phase unwrapping algorithms," IEEE Trans. on Image Process. **25**(6), 2601–2609 (2016).
4. L. An, Q.-S. Xiang, and S. Chavez, "A fast implementation of the minimum spanning tree method for phase unwrapping," IEEE Transactions on Medical Imaging **19**(8), 805–808 (2000).
5. T. J. Flynn, "Two-dimensional phase unwrapping with minimum weighted discontinuity," J. Opt. Soc. Am. A **14**(10), 2692–2701 (1997).
6. J. Xu, D. An, X. Huang, and P. Yi, "An efficient minimum-discontinuity phase-unwrapping method," IEEE Geosci. Remote Sensing Lett. **13**(5), 666–670 (2016).
7. D. C. Ghiglia, G. A. Mastin, and L. A. Romero, "Cellular-automata method for phase unwrapping," J. Opt. Soc. Am. A **4**(1), 267–280 (1987).
8. M. Servin, J. L. Marroquin, and F. J. Cuevas, "Demodulation of a single interferogram by use of a two-dimensional regularized phase-tracking technique," Appl. Opt. **36**(19), 4540–4548 (1997).
9. H. Y. H. Huang, L. Tian, Z. Zhang, Y. Liu, Z. Chen, and G. Barbastathis, "Path-independent phase unwrapping using phase gradient and total-variation (tv) denoising," Opt. Express **20**(13), 14075–14089 (2012).
10. K. O'Shea and R. Nash, "An introduction to convolutional neural networks," *CoRR* **abs/1511.08458** (2015).
11. M. Pak and S. Kim, "A review of deep learning in image recognition," in *2017 4th International Conference on Computer Applications and Information Processing Technology (CAIPT)*, (2017), pp. 1–3.
12. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," arXiv, arXiv:1512.03385v1 (2015).
13. K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv, arXiv:1409.1556v6 (2014).
14. C. Chen, C. Qin, H. Qiu, G. Tarroni, J. Duan, W. Bai, and D. Rueckert, "Deep learning for cardiac image segmentation: a review," Front. Cardiovasc. Med. **7**, 25 (2020).
15. A. Işın, C. Direkoğlu, and M. Şah, "Review of mri-based brain tumor image segmentation using deep learning methods," Procedia Comput. Sci. **102**, 317–324 (2016).
16. R. Wang, T. Lei, R. Cui, B. Zhang, H. Meng, and A. K. Nandi, "Medical image segmentation using deep learning: A survey," IET Image Process. **16**(5), 1243–1267 (2022).
17. X. Zhou, W. Gong, W. Fu, and F. Du, "Application of deep learning in object detection," in *2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS)*, (IEEE, 2017), pp. 631–634.
18. A. R. Pathak, M. Pandey, and S. Rautaray, "Application of deep learning for object detection," Procedia Comput. Sci. **132**, 1706–1717 (2018).
19. C. Tian, L. Fei, W. Zheng, Y. Xu, W. Zuo, and C.-W. Lin, "Deep learning on image denoising: An overview," Neural Networks **131**, 251–275 (2020).
20. K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," IEEE Trans. on Image Process. **26**(7), 3142–3155 (2017).
21. D. Liu, B. Wen, X. Liu, and T. S. Huang, "When image denoising meets high-level vision tasks: A deep learning approach," *CoRR* **abs/1706.04284** (2017).
22. K. Yan, L. Chang, M. Andrianakis, V. Tornari, and Y. Yu, "Deep learning-based wrapped phase denoising method for application in digital holographic speckle pattern interferometry," Appl. Sci. **10**(11), 4044 (2020).
23. S. Park, Y. Kim, and I. Moon, "Automated phase unwrapping in digital holography with deep learning," Biomed. Opt. Express **12**(11), 7064–7081 (2021).
24. F. Yang, T.-A. Pham, N. Brandenberg, M. P. Lütolf, J. Ma, and M. Unser, "Robust phase unwrapping via deep image prior for quantitative phase imaging," IEEE Trans. on Image Process. **30**, 7025–7037 (2021).
25. W. Yin, Q. Chen, S. Feng, T. Tao, L. Huang, M. Trusiak, A. Asundi, and C. Zuo, "Temporal phase unwrapping using deep learning," Sci. Rep. **9**(1), 20175 (2019).
26. G. E. Spoorthi, R. K. Sai Subrahmanyam Gorthi, and S. Gorthi, "Phasenet 2.0: Phase unwrapping of noisy data based on deep learning approach," IEEE Trans. on Image Process. **29**, 4862–4872 (2020).
27. K. Wang, Y. Li, Q. Kemao, J. Di, and J. Zhao, "One-step robust deep learning phase unwrapping," Opt. Express **27**(10), 15100–15115 (2019).
28. O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," arXiv, arXiv:1505.04597v1 (2015).
29. O. Oktay, J. Schlemper, L. L. Folgoc, M. C. H. Lee, M. P. Heinrich, K. Misawa, K. Mori, S. G. McDonagh, N. Y. Hammerla, B. Kainz, B. Glocker, and D. Rueckert, "Attention u-net: Learning where to look for the pancreas," *CoRR* **abs/1804.03999** (2018).
30. D. Jin, R. Zhou, Z. Yaqoob, and P. T. So, "Tomographic phase microscopy: principles and applications in bioimaging," J. Opt. Soc. Am. B **34**(5), B64–B77 (2017).

31. V. Balasubramani, A. Kuś, H.-Y. Tu, C.-J. Cheng, M. Baczewska, W. Krauze, and M. Kujawińska, "Holographic tomography: techniques and biomedical applications [invited]," Appl. Opt. **60**(10), B65–B80 (2021).
32. M. Takeda, H. Ina, and S. Kobayashi, "Fourier-transform method of fringe-pattern analysis for computer-based topography and interferometry," J. Opt. Soc. Am. **72**(1), 156–160 (1982).
33. A. Kuś, W. Krauze, P. L. Makowski, and M. Kujawińska, "Holographic tomography: hardware and software solutions for 3d quantitative biomedical imaging," ETRI Journal **41**(1), 61–72 (2019).
34. P. Machnio, M. Ziemczonok, and M. Kujawińska, "Reconstruction enhancement via projection screening in holographic tomography," Photonics Lett. Pol. **13**(2), 37–39 (2021).
35. Z. Malacara and M. Servin, *Interferogram analysis for optical testing* (CRC, 2018).
36. W. Krauze, A. Kuś, M. Ziemczonok, M. Haimowitz, S. Chowdhury, and M. Kujawińska, "3d scattering microphantom sample to assess quantitative accuracy in tomographic phase microscopy techniques," Sci. Rep. **12**(1), 19586 (2022).
37. P. Stępień, M. Ziemczonok, M. Kujawińska, M. Baczewska, L. Valenti, A. Cherubini, E. Casirati, and W. Krauze, "Numerical refractive index correction for the stitching procedure in tomographic quantitative phase imaging," Biomed. Opt. Express **13**(11), 5709–5720 (2022).
38. V. Balasubramani, M. Kujawińska, C. Allier, V. Anand, C.-J. Cheng, C. Depeursinge, N. Hai, S. Juodkazis, J. Kalkman, and A. Kuś, "Roadmap on digital holography-based quantitative phase imaging," J. Imaging **7**(12), 252 (2021).
39. X. Su and W. Chen, "Reliability-guided phase unwrapping algorithm: a review," Opt. Lasers Eng. **42**(3), 245–261 (2004).
40. C. Zuo, J. Qian, S. Feng, W. Yin, Y. Li, P. Fan, J. Han, K. Qian, and Q. Chen, "Deep learning in optical metrology: a review," Light: Sci. Appl. **11**(1), 39 (2022).
41. G. E. Spoorthi, S. Gorthi, and R. K. S. S. Gorthi, "Phasenet: A deep convolutional neural network for two-dimensional phase unwrapping," IEEE Signal Process. Lett. **26**(1), 54–58 (2019).
42. T. Zhang, S. Jiang, Z. Zhao, K. Dixit, X. Zhou, J. Hou, Y. Zhang, and C. Yan, "Rapid and robust two-dimensional phase unwrapping via deep learning," Opt. Express **27**(16), 23173–23185 (2019).
43. J. Zhang, X. Tian, J. Shao, H. Luo, and R. Liang, "Phase unwrapping in optical metrology via denoised and convolutional segmentation networks," Opt. Express **27**(10), 14903–14912 (2019).
44. G. Dardikman-Yoffe, D. Roitshtain, S. K. Mirsky, N. A. Turko, M. Habaza, and N. T. Shaked, "Phun-net: ready-to-use neural network for unwrapping quantitative phase images of biological cells," Biomed. Opt. Express **11**(2), 1107–1121 (2020).
45. G. Dardikman and N. T. Shaked, "Phase unwrapping using residual neural networks," in *Imaging and Applied Optics 2018 (3D, AO, AIO, COSI, DH, IS, LACSEA, LS & C, MATH, pcAOP)*, (Optica Publishing Group, 2018), p. CW3B.5.
46. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," Commun. ACM **63**(11), 139–144 (2020).
47. C. Li, Y. Tian, and J. Tian, "A method for single image phase unwrapping based on generative adversarial networks," in *Eleventh International Conference on Digital Image Processing (ICDIP 2019)*, vol. 11179 J.-N. Hwang and X. Jiang, eds., International Society for Optics and Photonics (SPIE, 2019), p. 1117911.
48. K. Wang, Q. Kemao, J. Di, and J. Zhao, "Deep learning spatial phase unwrapping: a comparative review," Adv. Photonics Nexus **1**(01), 014001 (2022).
49. N. Siddique, S. Paheding, C. P. Elkin, and V. Devabhaktuni, "U-net and its variants for medical image segmentation: A review of theory and applications," IEEE Access **9**, 82031–82057 (2021).
50. S. Sharma, S. Sharma, and A. Athaiya, "Activation functions in neural networks," towards data science **6**, 310–316 (2017).
51. K. P. Murphy, *Machine learning: a probabilistic perspective* (MIT, 2012).
52. B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," arXiv, arXiv:1505.00853v2 (2015).
53. S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," arXiv, arXiv:1502.03167v3 (2015).
54. A. F. Agarap, "Deep learning using rectified linear units (relu)," *CoRR abs/1803.08375* (2018).
55. M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," arXiv, arXiv:1508.04025v5 (2015).
56. N. Johnson, S. Kotz, and N. Balakrishnan, "Normal distributions," Continuous univariate distributions **1**, 156–157 (1987).
57. "BiOpto team website," http://biophase.pl. Accessed: 2022-10-19.
58. M. Baczewska, W. Krauze, A. Kus, P. Stepien, K. Tokarska, K. Zukowski, E. Malinowska, Z. Brzózka, and M. Kujawinska, "On-chip holographic tomography for quantifying refractive index changes of cells' dynamics," in *Quantitative Phase Imaging VIII*, vol. 11970 Y. Liu, G. Popescu, and Y. Park, eds., International Society for Optics and Photonics (SPIE, 2022), p. 1197008.
59. M. Ziemczonok, A. Kuś, P. Wasylczyk, and M. Kujawińska, "3d-printed biological cell phantom for testing 3d quantitative phase imaging systems," Sci. Rep. **9**(1), 18872 (2019).
60. M. Ziemczonok, A. Kus, and M. Kujawinska, "Optical diffraction tomography meets metrology - measurement accuracy on cellular and subcellular level," Measurement **195**, 111106 (2022).
61. D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv, arXiv:1412.6980v9 (2014).

62. K. Sumanth, V. Ravi, and R. K. Gorthi, “A multi-task learning for 2d phase unwrapping in fringe projection,” IEEE Signal Process. Lett. **29**, 797–801 (2022).
63. J. Zhang and Q. Li, “Eesanet: edge-enhanced self-attention network for two-dimensional phase unwrapping,” Opt. Express **30**(7), 10470–10490 (2022).
64. M. Gontarz, V. Dutta, M. Kujawinska, and W. Krauze, “Phase unwrapping using deep learning in holographic tomography - dataset,” Zenodo (2023). https://doi.org/10.5281/zenodo.7773979.
65. Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” IEEE Trans. on Image Process. **13**(4), 600–612 (2004).
66. A. Horé and D. Ziou, “Image quality metrics: Psnr vs. ssim,” in *2010 20th International Conference on Pattern Recognition*, (2010), pp. 2366–2369.