Fast phase processing in off-axis holography by CUDA including parallel phase unwrapping

Ohad Backoach,^{1,2} Saar Kariv,^{1,2} Pinhas Girshovitz,^{1} and Natan T. Shaked^{1,*}

^{1}*Department of Biomedical Engineering, Faculty of Engineering, Tel Aviv University, Tel Aviv 69978, Israel*
^{2}*Contributed equally*
**nshaked@tau.ac.il*

Abstract: We present parallel processing implementation for rapid extraction of the quantitative phase maps from off-axis holograms on the Graphics Processing Unit (GPU) of the computer using computer unified device architecture **(**CUDA) programming. To obtain efficient implementation, we parallelized both the wrapped phase map extraction algorithm and the two-dimensional phase unwrapping algorithm. In contrast to previous implementations, we utilized unweighted least squares phase unwrapping algorithm that better suits parallelism. We compared the proposed algorithm run times on the CPU and the GPU of the computer for various sizes of off-axis holograms. Using the GPU implementation, we extracted the unwrapped phase maps from the recorded off-axis holograms at 35 frames per second (fps) for 4 mega pixel holograms, and at 129 fps for 1 mega pixel holograms, which presents the fastest processing framerates obtained so far, to the best of our knowledge. We then used common-path off-axis interferometric imaging to quantitatively capture the phase maps of a micro-organism with rapid flagellum movements.

OCIS codes: (090.1995) Digital holography; (090.5694) Real-time holography; (100.5070) Phase retrieval; (100.5088) Phase unwrapping; (110.3175) Interferometric imaging; (120.3180) Interferometry.

References and links

1. B. Rappaz, A. Barbul, Y. Emery, R. Korenstein, C. Depeursinge, P. J. Magistretti, and P. Marquet, "Comparative study of human erythrocytes by digital holographic microscopy, confocal microscopy, and impedance volume analyzer," Cytometry A **73**(10), 895–903 (2008).
2. P. Girshovitz and N. T. Shaked, "Generalized cell morphological parameters based on interferometric phase microscopy and their application to cell life cycle characterization," Biomed. Opt. Express **3**(8), 1757–1773 (2012).
3. B. Kemper, P. Langehanenberg, and G. von Bally, "Digital holographic microscopy: A new method for surface analysis and marker-free dynamic life cell imaging," Optik Photonik **2**(2), 41–44 (2007).
4. S. Grilli, V. Vespini, F. Merola, P. Ferraro, L. Miccio, M. Paturzo, and S. Coppola, "Exploring the capabilities of digital holography as tool for testing optical microstructures," 3D Res. **2**, 1007 (2011).
5. S. Gawad, M. Giugliano, M. Heuschkel, B. Wessling, H. Markram, U. Schnakenberg, P. Renaud, and H. Morgan, "Substrate arrays of iridium oxide microelectrodes for in vitro neuronal interfacing," Front. Neuroeng. **2**, 1 (2009).
6. C. Edwards, A. Arbabi, G. Popescu, and L. L. Goddard, "Optically monitoring and controlling nanoscale topography during semiconductor etching," Light Sci. Appl. **30**, e130 (2012).
7. P. Girshovitz and N. T. Shaked, "Real-time quantitative phase reconstruction in off-axis digital holography using multiplexing," Opt. Lett. **39**(8), 2262–2265 (2014).
8. P. Girshovitz and N. T. Shaked, "Fast phase processing in off-axis holography using multiplexing with complex encoding and live-cell fluctuation map calculation in real-time," Opt. Express **23**(7), 8773–8787 (2015).
9. H. Pham, H. Ding, N. Sobh, M. Do, S. Patel, and G. Popescu, "Off-axis quantitative phase imaging processing using CUDA: toward real-time applications," Biomed. Opt. Express **2**(7), 1781–1793 (2011).
10. M. Habaza, B. Gilboa, Y. Roichman, and N. T. Shaked, "Tomographic phase microscopy with 180° rotation of live cells in suspension by holographic optical tweezers," Opt. Lett. **40**(8), 1881–1884 (2015).
11. K. Kim, K. S. Kim, H. Park, J. C. Ye, and Y. Park, "Real-time visualization of 3-D dynamic microscopic objects using optical diffraction tomography," Opt. Express **21**(26), 32269–32278 (2013).

#251845 Received 14 Oct 2015; revised 6 Jan 2016; accepted 7 Jan 2016; published 8 Feb 2016

Corrected: 16 February 2016

12. Y. Jang, J. Jang, and Y. Park, "Dynamic spectroscopic phase microscopy for quantifying hemoglobin concentration and dynamic membrane fluctuation in red blood cells," Opt. Express **20**(9), 9673–9681 (2012).
13. M. Mir, K. Tangella, and G. Popescu, "Blood testing at the single cell level using quantitative phase and amplitude microscopy," Biomed. Opt. Express **2**(12), 3259–3266 (2011).
14. Z. Wang, K. Tangella, A. Balla, and G. Popescu, "Tissue refractive index as marker of disease," J. Biomed. Opt. **16**(11), 116017 (2011).
15. C. Edwards, A. Arbabi, G. Popescu, and L. L. Goddard, "Optically monitoring and controlling nanoscale topography during semiconductor etching," Light Sci. Appl. **1**(9), e30 (2012).
16. R. Zhou, C. Edwards, A. Arbabi, G. Popescu, and L. L. Goddard, "Detecting 20 nm wide defects in large area nanopatterns using optical interferometric microscopy," Nano Lett. **13**(8), 3716–3721 (2013).
17. L. Waller, CalOptrics: Compuational Optical Imaging Open Source Library for CUDA, UC Berkeley (2014). *https://github.com/Waller-Lab/CalOptrics*
18. A. Doronin and I. Meglinski, "Online object oriented Monte Carlo computational tool for the needs of biomedical optics," Biomed. Opt. Express **2**(9), 2461–2469 (2011).
19. P. A. Karasev, D. P. Campbell, and M. A. Richards, "Obtaining a 35x speedup in 2d phase unwrapping using commodity graphics processors," in 2007 IEEE Radar Conference (IEEE, 2007), pp. 574–578.
20. P. Mistry, S. Braganza, D. Kaeli, and M. Leeser, "Accelerating phase unwrapping and affine transformations for optical quadrature microscopy using CUDA," in 2nd Workshop on General Purpose Processing on Graphics Processing Units (ACM, Washington, D.C., 2009), pp. 28–37.
21. D. C. Ghihlia and M. D. Pritt, Two-dimensional Phase Unwrapping: Theory, Algorithms, and Software (Wiley, 1998).
22. University of Oslo, "Implementation of the DFT and the DCT," in MAT-INF2360: Applications of Linear Algebra (2012). *http://www.uio.no/studier/emner/matnat/math/MAT-INF2360/v12/fft.pdf*
23. P. Girshovitz and N. T. Shaked, "Compact and portable low-coherence interferometer with off-axis geometry for quantitative phase microscopy and nanoscopy," Opt. Express **21**(5), 5701–5714 (2013).

1. Introduction

Off-axis digital holography captures, in a single camera exposure, an interference pattern between a light beam interacting with a sample and a reference beam, where both beams interfere at a small angle on the camera. From this single off-axis hologram, the complex wave front of the light interacted with the sample can be extracted. This wave front contains both the amplitude and the phase profiles of the sample. The phase profile is of particular interest for samples that induce negligible amplitude change, and thus can be imaged in a good quality only through their phase. Per each spatial point, the acquired phase is proportional to the optical path delay between the sample and the reference beams, which accounts for both the refractive index and the geometrical path delays in the sample beam path. The latter can be used to obtain the two-dimensional (2-D) thickness map of the sample. Common applications include label-free, quantitative imaging of optically transparent biological cells *in vitro* [1–3] and nondestructive metrology of thin elements [3–6].

The quantitative phase extraction process from an off-axis hologram is performed digitally, and includes spatial filtering by two 2-D Fourier transforms, and 2-D phase unwrapping to solve 2π ambiguities in spatial points, where the optical path delay is larger than the illumination wavelength. Typically, when using Matlab and the CPU of a conventional personal computer utilizing a single processing core, extracting the unwrapped phase map from one mega pixel hologram can take half a second [7–9], which preludes real-time processing and visualization. This limitation becomes even more critical as the number of pixels in the processed hologram increases.

Fast holographic processing and visualization is specifically important for real-time clinical decisions, for example, for analyzing cells in a flow cytometer. Alternatively in optical metrology, fast holographic processing and visualization is useful for feedback to the lithography machine during short etching processes.

Fast holographic processing is also required for processing a large amount of holographic data within a reasonable amount of time, for example, for tomographic phase microscopy, where many interferometric projections are acquired from various points of view and then processed to the three-dimensional (3-D) refractive-index map of the sample [10,11], or for spectroscopic interferometry, when many holograms are acquired in various wavelengths [12]. Additionally, fast holographic processing is required even for stationary samples, not during dynamic processes, when the sample is scanned to obtain extended field of view, for example, for broad field of view quantitative phase imaging of a blood smear [13], thin histological tissues [14], etched elements [15], or silicon wafers [16].

To allow fast off-axis hologram processing, we have lately proposed new algorithms that can extract the unwrapped phase maps from 1 mega pixel off-axis holograms at up to 45 frames per second (fps) by using a single processing unit of a conventional computer [7,8]. This is obtained by utilizing the Fourier-transform-based holographic processing more efficiently; for example, by multiplexing several off-axis holograms together and applying a single Fourier transform to process all of them together. The ability to process holograms in more than video rate of 25 fps enables performing additional real-time calculations, such as obtaining the fluctuation map of the sample in real time [8].

However, when the hologram size increases, real-time processing is not possible on a regular computer anymore. For example, for 4 mage pixel holograms, our fastest algorithm in Ref [8]. can obtain less than 10 fps on the CPU. In this case, other computational platforms are required. The graphic processing unit (GPU) of a conventional computer contains many processing units, so that the overall calculation can be divided to smaller calculations, each is performed on one of the GPU's multiple internal processing units in parallel, while speeding up the total calculation time. Compute unified device architecture (CUDA) is a parallel programming environment and application programming interface that allows relatively easy programming implementations on NVidia's GPU. In the optical engineering field, CUDA was previously used to speed up various optical processing tasks [17], including Monte Carlo simulations for light-tissue interaction [18], phase unwrapping [19,20], and others.

Specifically for off-axis holographic processing, Ref [9]. suggests extracting the unwrapped phase maps on the GPU of the computer, as implemented in CUDA, while using the conventional Fourier-based algorithm and an unwrapping algorithm that utilizes the Goldstein’s branch-cut method. This algorithm, however, contains many sequential operations, especially in its branch-cut placement stage and it does not suit parallelism. Specifically, for noisy images, when the number of residues increases, the branch-placement calculation time will increase markedly. In Ref [9], the framerate obtained for the phase unwrapping process was 40.7 fps for 1 mega pixel holograms, implying at least 4 times framerate decrease for 4 mega pixel holograms.

In the current paper, we present an efficient CUDA implementation for extracting of the quantitative phase maps from off-axis holograms on the GPU using an improved Fourier-based algorithm and a phase unwrapping algorithm that better suits parallelism. Using this implementation, we obtained unwrapped phase map extraction at 129 fps for 1 mega pixel holograms (in comparison to 40.7 fps in the previous implementation), and 35 fps for 4 mega pixel holograms (more than video rate).

We first shortly review the entire Fourier-based algorithm for the extraction of phase maps from the off-axis holograms. Then, we present the parallel CUDA implementation of this algorithm. Next, we present the experimental setup we used with for the experimental demonstrations, and the experimental results obtained.

2. Phase extraction from off-axis holograms

Assuming straight off-axis fringes across the horizontal axis *m*, per each spatial point *(m, n)*, the off-axis hologram recorded by the camera can be expressed as follows:

$$\begin{aligned} H &= \left|E_s + E_r\right|^2 = \left|E_s\right|^2 + \left|E_r\right|^2 + E_s^* E_r + E_s E_r^* \\ &= \left|E_s\right|^2 + \left|E_r\right|^2 + 2\left|E_s\right|\left|E_r\right|\cos\left(\varphi - k_m \sin\left(\theta\right)\right), \end{aligned} \tag{1}$$

where E_s and E_r are the complex wave fronts of the sample and the reference beams, respectively, φ is the phase of the sample (assuming a plane-wave reference), which is related to the optical path delay (or the optical thickness) of the sample as follows $OPD = 2\pi\varphi / \lambda$, where λ is the illumination wavelength, $k_m = 2\pi m / \lambda$ is the spatial frequency of the fringes, and θ is the angle between the sample and the reference waves. H, E_s, E_r and OPD are functions of the transverse coordinate (m, n). In the spatial-frequency domain, the Fourier-transforms of $E_s^* E_r$ and $E_s E_r^*$, referred to as the cross-correlation terms, are located on different sides of the spatial-frequency domain, so that any one of them can be chosen and analyzed to the complex wave front of the sample. Figure 1 presents the steps of the entire phase map reconstruction algorithm. First, in step A1, we convert the digital off-axis hologram recorded by the camera, containing $N \times N$ real-value pixels, to the spatial-frequency domain using a 2-D FFT. Then, in step A2, we crop one of the cross-correlation terms, containing $N/4 \times N/4$ complex-value pixels, which contains the entire wave-front spatial-frequency content, provided that the optical setup is well aligned and the off-axis angle between the beams induces enough separation between the central auto-correlation term and the cross-correlation terms [7]. Next, in step A3, we transform the cropped cross-correlation term back to the image domain by using a 2-D IFFT, resulting in complex matrix, containing $N/4 \times N/4$ pixels, and representing the sample wave front. In step A4, we correct for stationary aberrations and curvatures in the beam profile by dividing the sample wave front from step A3 by another wave front obtained in advance without the sample presence. Afterwards, in the step A5, we take the phase argument of the resulting complex matrix and perform 2-D phase unwrapping to solve 2π ambiguities in the phase map of the sample. Section 3.2 presents the chosen unwrapping algorithm, which suits parallelism with CUDA. Finally, if needed, we can enlarge the unwrapped phase map, containing $N/4 \times N/4$, to the final unwrapped phase matrix containing $N \times N$ pixels. This resizing step, however, does not add new information to the final image.

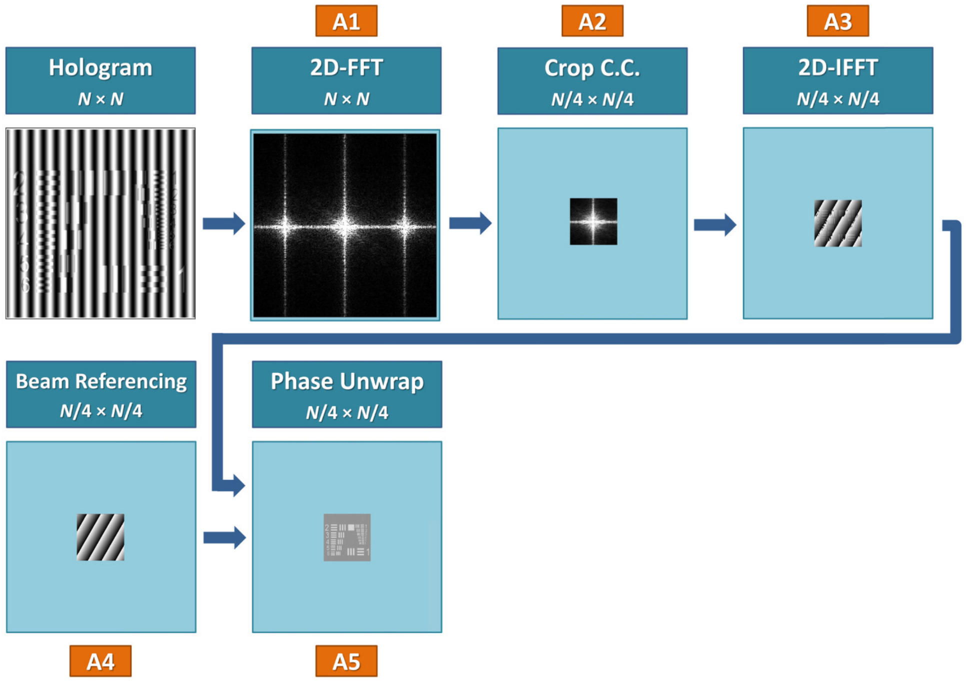

Fig. 1. Digital process for the extraction of the phase map from an off-axis image hologram.

3. Implementation on the GPU

In general, parallelizing a multi-step algorithm might be a challenging task; while some of the steps can be parallelized easily, other steps might be fundamentally sequential and impossible to parallelize. Therefore, one must take a great care in every step of the algorithm to ensure efficient implementation. In case there are several options for algorithm selection, for example in our case the phase unwrapping algorithm, non-sequential algorithms should be preferred to algorithms that contain sequential steps.

The GPU architecture consists thousands of small cores designed for handling multiple tasks simultaneously and in a parallel way. CUDA allows software developers to utilize the GPU for processing tasks more easily. Below, we first discuss the GPU implementation for extraction of the wrapped phase map from an off-axis hologram, and then present the parallel implementation of the chosen 2-D phase unwrapping algorithm, in comparison to the previously one used.

3.1 Wrapped phase extraction from an off-axis hologram on the GPU

Steps A1 – A3 in Fig. 1 can be straightforwardly implemented in parallel on the GPU by processing all the pixels concurrently, since each of the pixels can be calculated without dependency on any other pixel. Steps A1 and A3, the 2D FFT and 2D IFFT, are computed using built-in CUDA functions. In step A2, one of the cross-correlation terms is cropped from the spatial-frequency domain so that its maximum value is in the middle of the frame. The location of the maximum value is found in advance by a serial sorting algorithm that runs only once per hologram set with a certain carrier fringe frequency and orientation. The cropping dimensions are defined using a designated function that finds the closest $N/4$ integer, which is also a power of 2. This operation can also be done once in advance per hologram set. The cropping itself is done per each hologram processed in the set and in parallel on the GPU, while already using the known cross-correlation term position and cropping size. The beam referencing stage in step A4 is also done in parallel on the GPU (pixel by pixel), by dividing the complex matrix resulting from step A3 with the sample-free complex matrix, calculated once per a hologram set. The phase argument of the resulting complex matrix is the wrapped phase map, which is retrieved from the complex matrix, pixel by pixel, by a GPU function implementing 4 quadrant arctangent. This phase map should be unwrapped to solve 2π ambiguities in all spatial areas where the optical path delay is larger than the wavelength used.

3.2 Phase unwrapping on the GPU

The 4 quadrant arctangent function yields the wrapped phase matrix $\psi(m,n)$ with values confined to $(-\pi,\pi]$, which is mathematically defined as follows:

$$\psi(m,n) = \varphi(m,n) + 2\pi q(m,n), \tag{2}$$

where $q(m,n)$ is an integer function that forces $-\pi < \psi(m,n) \le \pi$. Thus, $\psi(m,n)$ is a nonlinear function of the actual phase $\varphi(m,n)$. The phase unwrapping process eliminates the non-continuous nature of $\psi(m,n)$, which occurs if the sample optical path delay is larger than the wavelength of light λ; thereby reconstructing the actual phase map $\varphi(m,n)$.

Reconstruction of the actual phase map is the most intense in terms of computational resources. For this reason, one needs to carefully choose the right 2-D phase unwrapping algorithm and design the best way of implementing it on the chosen computational platform.

In general, there are two main families of methods for 2-D phase unwrapping: path following methods and minimum norm methods [21]. In the path following methods, the new unwrapped matrix is built by integrating phase differences around a certain pixel on a certain path *C*:

$$\varphi(r) = \varphi(r_0) + \int_c \nabla \varphi \cdot dr, \tag{3}$$

where $r = \sqrt{m^2 + n^2}$ is the radial distance from the pixel, and r_0 is the starting point. The result of this integration might be path dependent due to noise and aliasing in the input matrix. To avoid this and choose independent paths, the path following methods include internal steps for balancing of noisy points and summing of pixels for integration, steps which are impossible to parallelize.

The Goldstein's phase unwrapping algorithm used in Ref [9] is a path following method. This algorithm contains three steps: (a) Residue identification, which marks a pixel as positive residue if the integral over a closed four pixel loop is greater than zero. If the integral is lower than zero, the pixel residual is marked as negative. Otherwise, in case of a zero result of the integral, the pixel will be residue free; (b) Branch-cut placement, which uses enlarging and searching over a search box on the image, while computing the charge cumulatively. This step requires knowledge on the other residues and whether they are branch cut pixels or they are connected to other residues. (c) Unwrapping around branch cuts, which requires that one of the neighboring pixels is already unwrapped. Step (b) is sequential in nature and impossible to parallelize. Step (c) is hard to parallelize since it requires pixel status synchronization. The number of residues is dependent on the quality of the wrapped phase map. For an increased number of residues, the Goldstein's algorithm implementation might be computationally heavy due to the large number of sequential operations.

Minimum norm methods are, in general, more suitable for parallelism. Specifically, in the minimum norm methods with least squares error (L^2 norm), we seek the unwrapped phase whose local derivatives match the measured derivatives as closely as possible. These methods use the least squares approach, meaning that the sum (integral) of the squared differences between the gradient of the solution and that of the measurements is minimized. Mathematically, we want to find φ that minimizes the following function:

$$J = \iint \left|\varphi_x - \psi_x\right|^2 + \left|\varphi_y - \psi_y\right|^2 dxdy, \tag{4}$$

where φ_x and φ_y are the derivatives of the unwrapped phase along the horizontal and vertical axes, respectively, and ψ_x and ψ_y are the derivatives of the wrapped phase along the horizontal and vertical axes, respectively. To obtain the appropriate φ that minimizes this function, it was shown that the following partial differential equation needs to be solved [21]:

$$\frac{\partial}{\partial x}(\varphi_x - \psi_x) + \frac{\partial}{\partial y}(\varphi_y - \psi_y) = 0, \tag{5}$$

which yields:

$$\varphi_{xx} + \varphi_{yy} = \psi_{xx} + \psi_{yy}, \tag{6}$$

where φ_{xx} and φ_{yy} are the second derivatives of the unwrapped phase along the horizontal and vertical axes, respectively, and ψ_{xx} and ψ_{yy} are the second derivatives of the wrapped phase along the horizontal and vertical axes, respectively. Then, the following Poisson's equation has to be solved:

$$\nabla^2 \varphi = \rho, \tag{7}$$

where φ is the unwrapped phase solution and $\rho = \psi_{xx} + \psi_{yy}$ is the wrapped phase second derivative summation on both axes.

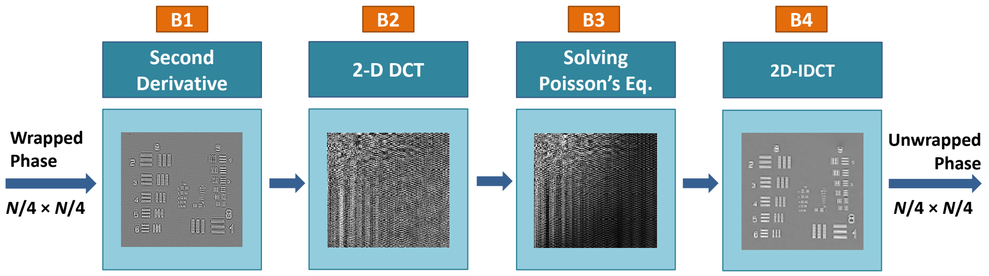

Fig. 2. The DCT-based UWLS 2-D phase unwrapping algorithm.

We are dealing with the discrete case, in which we want to find $\varphi(m,n)$ and $\rho(m,n)$ is known. To find this solution, we chose the discrete cosine transform (DCT) based un-weighted least squares (UWLS) algorithm. In this phase unwrapping algorithm, all the internal steps can be implemented in parallel, in contrast to the Goldstein's algorithm, and thus can be utilized efficiently on the GPU.

After applying a DCT transform, the Poisson's equation of Eq. (7) becomes:

$$\left[2\cos\left(\frac{\pi m}{M}\right)+2\cos\left(\frac{\pi n}{N}\right)-4\right]DCT\{\varphi(m,n)\}=DCT\{\rho(m,n)\}, \tag{8}$$

which is a linear equation. Therefore, we can find the unwrapped phase $\varphi(m,n)$ as follows:

$$\varphi(m,n)=IDCT\left\{DCT\{\rho(m,n)\}/\left[2\cos\left(\frac{\pi m}{M}\right)+2\cos\left(\frac{\pi n}{N}\right)-4\right]\right\}, \tag{9}$$

Solving this linear equation can be done in parallel since each pixel can be calculated without dependency on the other pixels. In CUDA, we implemented the DCT and the inverse DCT (IDCT) transforms based on a CUDA library for FFTs and the DCT-FFT relations given in Ref [22].

Figure 2 summarizes the steps of this unwrapping algorithm. As shown in this figure, the algorithm includes applying a second derivative on the wrapped phase map (step B1), applying a 2-D DCT on the result (step B2), solving the frequency-transformed Poisson's equation (step B3), and applying a 2-D IDCT on the solution (step B4), which yields the final unwrapped phase.

4. Experimental setup

The experimental setup used for the demonstrations could be any interferometer creating off-axis holograms on the camera. In this paper, we used the interferometric system shown in Fig. 3. This figure presents the off-axis τ interferometer [23], which is a close-to-common-path off-axis imaging interferometer positioned at the output port of a microscope illuminated by a plane wave of coherent or partially coherent light (HeNe of 632.8 nm wavelength, or supercontinuum laser source plus AOTF, with 6.4 nm spectral bandwidth around 514 nm). In this module, the image plane at the output of the microscope is Fourier transformed by lens L4. The beam then splits into two beams via a beam splitter. One of the beams, referred to as the sample beam, is reflected via retroreflector RR back to the beam splitter and splits again towards lens L5, which Fourier transforms it back to a the camera sensor at a small off-axis angle. The other beam, referred to as the reference beam, is spatially filtered via pinhole P2.

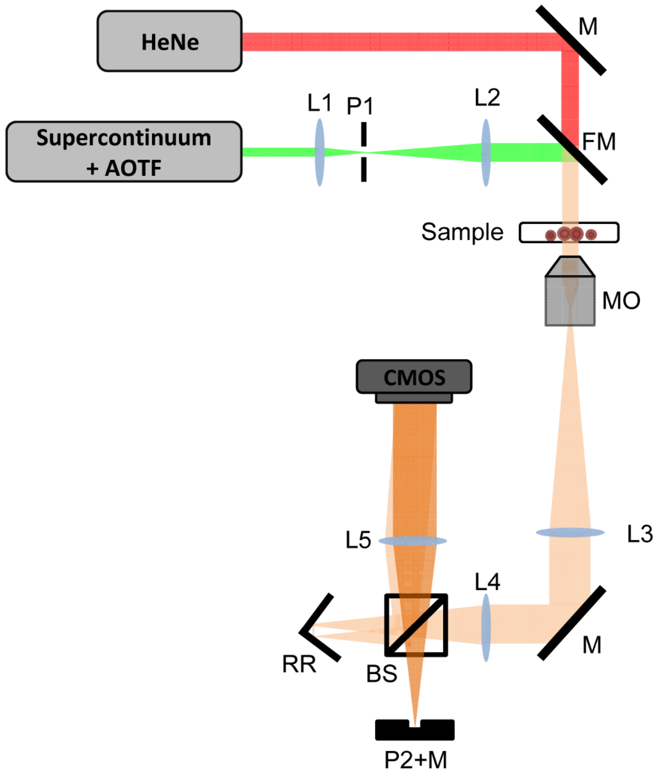

Fig. 3. The off-axis imaging interferometer used for the hologram acquisition. HeNe – Helium-Neon laser; AOTF – Acousto-optical tunable filter; L1-L5 – lenses; P1,P2 – Pinholes; M – Mirror; FM – Flip mirror, for light source selection, MO – Microscope objective; RR – Retro-reflector; CMOS – Digital camera.

The filtering erases the sample high-spatial frequencies and thereby effectively creates a reference beam only after the exit of the microscope. After passing through the pinhole, the beam is reflected by a mirror back to the beam splitter, propagates through lens L5 and Fourier transformed back to the image plane on the camera sensor, while interfering with the sample beam, creating an off-axis hologram on the camera. The angle between the sample and the reference beams is chosen so that there are three pixels per interference cycle, ensuring an optimal separation between the auto-correlation and the cross-correlation terms.

For the experiments, we used 63×1.4-NA oil-immersion microscope objective, achromatic lenses with focal lengths of 30, 50, 150, 100 and 200 mm for L1, L2, L3, L4 and L5, respectively, and a fast monochromatic camera (Grasshopper3, GS3-U3-23S6M-C, Point grey), containing 1920 × 1200 square pixels of 5.86 µm each.

5. Experimental results

5.1 Evaluation of the quality of the reconstruction

To evaluate the quality of the reconstruction using the DCT-based UWLS 2-D phase unwrapping algorithm, we used focused ion beam lithography to create an optically transparent phase target, which is based on a 1951 USAF resolution test target mask. We then used the optical system with the supercontinuum/AOTF source shown in Fig. 3 to record the off-axis image hologram shown in Fig. 4(a). We used NVidia's GeForce GTX 650 GPU of a personal desktop computer with Intel Xeon E5-1603 2.8GHz 8GB RAM CPU. We used single-precision floating-point format. To compare the reconstruction quality, we first implemented the two 2-D unwrapping phase reconstruction algorithms discussed above, the

Goldstein's and DCT-based UWLS algorithms, on CPU-Matlab platforms. The results of the unwrapped phase map reconstructions are shown in Figs. 4(b) and (c). Figure 4(d) shows the reconstruction obtained by the UWLS algorithm implemented on the GPU, as explained in Section 3. As can be seen from Figs. 4(b-d) and from the cross sections across group 8, shown in Fig. 4(e) (marked with a black vertical line in Figs. 4(b-d)), all unwrapped phase maps present very similar reconstruction qualities. Thus, since the UWLS phase unwrapping algorithm can be more easily parallelized, it should be preferred when implemented on the GPU. Furthermore, although not demonstrated here, for noisy images, where the number of residues in the Goldstein's algorithm increases, the branch-placement calculation time increases significantly and the unwrapped phase quality might decrease. This disadvantage does not exist in the UWLS algorithm used in the current paper.

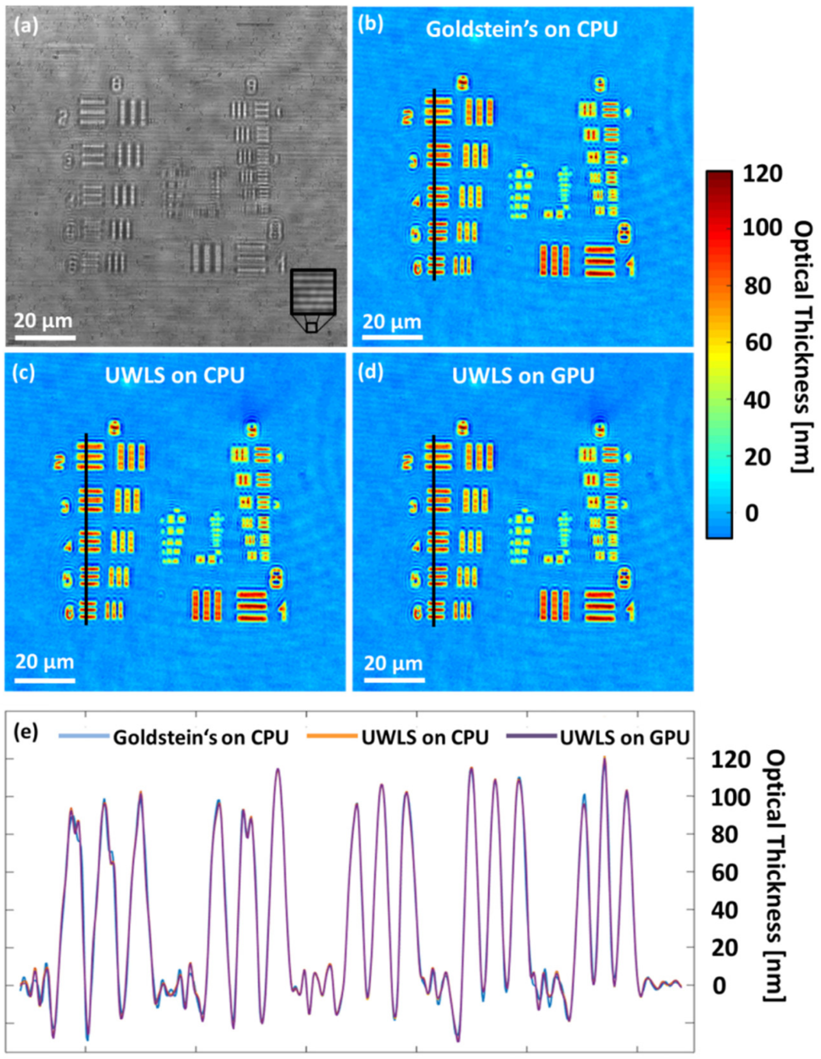

Fig. 4. Evaluation of the reconstruction quality. (a) Off-axis image hologram of a 1951 USAF phase test target. (b-d) The unwrapped phase maps reconstructed from the hologram on: (b) the CPU (Matlab) using the Goldstein's phase unwrapping algorithm, (c) the CPU (Matlab) using the UWLS algorithm, and (d) the GPU (CUDA) using the DCT-based UWLS phase unwrapping algorithm. (e) Cross section across group 8 as indicated by the black lines marked on Figs. 4(b-d).

5.2 Comparison of the processing times and framerates

To evaluate the processing time and framerate increase obtained by the proposed method, we processed off-axis holograms of 256 × 256, 512 × 512, 1024 × 1024 and 2048 × 2048 pixels. On the CPU, we implemented the process on Matlab and on C++ (with modern FFT/DCT libraries), separately. On the GPU, we implemented the process in parallel using CUDA. All implementations used the algorithms described in Figs. 1 and 2. Ten runs have been performed per each case. Table 1 summarizes the averaged processing times of the different steps for the various hologram sizes. The overall framerate for the different hologram sizes is presented in Fig. 5. As expected, the GPU processing time is much shorter, when compared to the CPU (C++) processing time, where the processing time increases with frame size, in both GPU and CPU implementations. As can be seen, we have reached speed factors of 6.8×, 6×, 4.1×, and 4.6× for 2048 × 2048, 1024 × 1024, 512 × 512 and 256 × 256 mega pixel holograms respectively. Similarly, when comparing the GPU processing times to the CPU (Matlab) processing time, we have reached speed factors of 7.3×, 5.3×, 6.3×, and 10.1× for 2048 × 2048, 1024 × 1024, 512 × 512 and 256 × 256 mega pixel holograms, respectively. To our knowledge, the off-axis hologram processing times achieved by the proposed method are the fastest currently exist, and specifically, this is the first time that more than video rate has been obtained for processing 4 mega pixel off-axis holograms.

Table 1. Comparison of the calculation times (in ms) on the CPU (C++) and the GPU (CUDA) of the various stages of the proposed implementation for various hologram sizes.

Step duration [ms]	2048 × 2048		1024 × 1024		512 × 512		256 × 256	
	CPU	GPU	CPU	GPU	CPU	GPU	CPU	GPU
Copy data in/out	-	5.93	-	1.91	-	0.15	-	0.07
Extract wrapped phase	61.5	16.27	16.7	4.29	2.8	1.10	1.2	0.30
Unwrap Phase	133.71	6.50	29.71	1.58	4.18	0.45	1.13	0.17
Total processing time	195.21	28.70	46.41	7.78	6.98	1.70	2.33	0.51

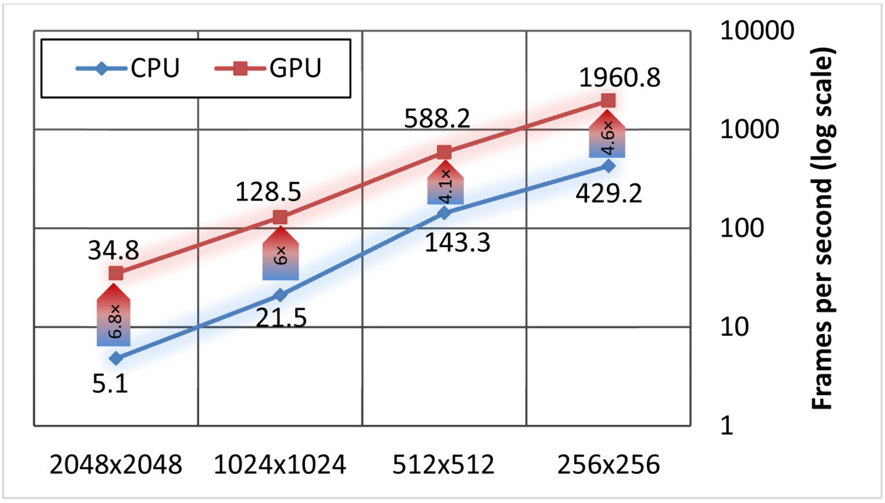

Fig. 5. Comparison of the framerates on the CPU (C + +) and the GPU of the entire reconstruction process of the proposed implementation, for various hologram sizes.

5.3 Micro-organism rapid quantitative phase imaging

To demonstrate the proposed GPU-implemented phase extraction method, we used it to acquire dynamic micro-organism, unicellular flagellate protist called *Euglena gracilis*, swimming in water and trapped between two glass slides. In this case, the HeNe was used due to the thickness of the sample. Figure 6 shows a single frame taken from a video showing the micro-organism quantitatively imaged with an imaging framerate of 129 fps (for dynamics see Visualization 1), as processed from 1024 × 1024 pixel hologram. Note that for better visualization in this video, we digitally decreased the presentation framerate; hence, we present this video slower than the actual framerate, appearing in the time stamp on the top left. In this video, one can see the complex and very rapid 3-D wavy movements of the thin flagellum.

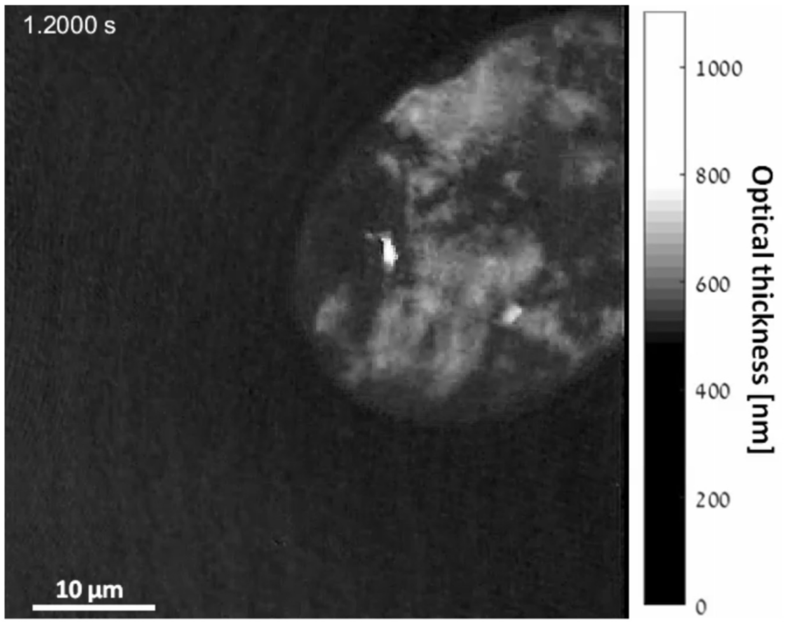

Fig. 6. Quantitative phase map of a micro-organism in water, as processed on the GPU using the proposed algorithm. Video of the rapid flagellum dynamics in 129 actual fps is shown in Visualization 1 (MP4, 1.4MB).

6. Conclusions

We have presented an efficient GPU implementation of unwrapped phase map extraction from off-axis image holograms. Since we used an improved Fourier-based algorithm and a phase unwrapping algorithm that suits parallelism, our GPU implementation is significantly faster compared to any previous one. We have obtained 129 fps for 1 mega pixel holograms, and, in the first time, we have obtained more than video rate, 35 fps, for 4 mega pixel holograms. The potential of this technique is for real-time extraction and quantitative visualization of the phase maps of thin object fast dynamics, or for obtaining quantitative imaging of large samples, while enabling obtaining full scans of these samples much faster. In addition, our technique is expected to find uses in tomographic phase microscopy and spectroscopic phase microscopy, in which a large number of off-axis holograms are acquired per each instance of the sample. Although our experimental demonstrations were mainly dedicated to imaging thin biological samples in transmission modes, our fast processing technique is expected to also find uses in metrology of thin elements during fast lithography processes in both transmission and reflection modes.

Acknowledgments

This research was supported by FP7 Marie Curie Career Integration Grant (CIG) No. 303559. We thank Dr. Tamir Gabay and Gili Dardikman from the OMNI group in Tel Aviv University for useful remarks.