J. Imaging **2015**, *1*, 31-44; doi:10.3390/jimaging1010031

OPEN ACCESS
Journal of
Imaging
ISSN 2313-433X
www.mdpi.com/journal/jimaging

Article

Fast Fourier-Based Phase Unwrapping on the Graphics Processing Unit in Real-Time Imaging Applications

Sam Van der Jeught [1,*], Jan Sijbers [2] and Joris J. J. Dirckx [1]

[1] Laboratory of Biomedical Physics, University of Antwerp, Groenenborgerlaan 171, B-2020 Antwerp, Belgium; E-Mail: joris.dirckx@uantwerpen.be

[2] iMinds-VisionLab, Department of Physics, University of Antwerp, Universiteitsplein 1, B-2610 Wilrijk, Belgium; E-Mail: jan.sijbers@uantwerpen.be

* Author to whom correspondence should be addressed; E-Mail: sam.vanderjeught@uantwerpen.be; Tel.: +32-(0)3-265-3553; Fax: +32-(0)3-265-3318.

Academic Editor: Gonzalo Pajares Martinsanz

Received: 30 March 2015 / Accepted: 10 June 2015 / Published: 15 June 2015

Abstract: Numerous imaging techniques measure data that are mathematically wrapped to the finite interval $[-\pi, \pi]$, corresponding to the principle value domain of the arctangent function. A wide range of reconstruction algorithms has been developed to obtain the true, unwrapped phase by adding an integral multiple of 2π to each point of the wrapped grid. However, the phase unwrapping procedure is hampered by the presence of noise, phase vortices or insufficiently sampled digital data. Unfortunately, reliable phase unwrapping algorithms are generally computationally intensive and their design often requires multiple iterations to reach convergence, leading to high execution times. In this paper, we present a high-speed phase unwrapping algorithm that is robust against noise and phase residues. By executing the parallel implementation of a single-step Fourier-based phase unwrapping algorithm on the graphics processing unit of a standard graphics card, we were able to reduce the total processing time of the phase unwrapping algorithm to < 5 ms when executed on a 640 × 480-pixel input map containing an arbitrarily high density of phase jumps. In addition, we expand upon this technique by inserting the obtained solution as a preconditioner in the conjugate gradient technique. This way, phase maps that contain regions of low-quality or invalid data can be unwrapped iteratively through weighting of local phase quality.

Keywords: real-time; phase unwrapping; graphics card; GPU

1. Introduction

Phase maps are generated in many applications in medicine, physics and engineering to quantify a wide variety of different physical properties. In optics, profilometric [1,2] and interferometric [3–5] techniques deliver phase maps to measure topological or mechanical properties such as shape, strain and local deformation of materials. Also, beyond the field of optics, phase images are generated by many imaging techniques. In magnetic resonance imaging, susceptibility-weighted phase mapping offers a new form of contrast enhancement between structures with different magnetic susceptibilities [6] and blood flow trajectories can be monitored through velocity encoded phase gradient echo imaging [7]. Remote sensing techniques that rely on synthetic aperture radar generate phase information that is used in a host of geodetic applications to monitor geodynamical phenomena such as tectonic deformation [8], volcanic activity [9] and glacial motion [10,11]. Electron holography recovers the electron phase to produce electrostatic and magnetostatic potential distribution maps of the sample [12,13] and optical holography techniques capture the phase of reflected light beams to measure relative movement or deformation [14,15] of the sample.

A common obstacle in the image processing schemes of the above techniques is the fact that the obtained phase map is defined only to within a single 2π-cycle. That is, the value of each measurement point in the phase map is generally wrapped to the region $[-\pi, \pi]$, corresponding to the principle value domain of the arctangent function. Therefore, phase unwrapping needs to be carried out before the phase map can be linked to the actual physical property it represents. The ability to correctly extract the true phase map from the principle value of the phase is severely complicated by the presence of noise and phase vortices [16–18]. To this end, a lot of effort has been made to increase the robustness of the phase-unwrapping process. When the image is processed sequentially, the phase unwrapping problem is a cumulative process where the error generated by the correction for a false jump or the miss of a genuine one will propagate throughout the rest of the image. In order to avoid or at least minimize false phase jumps, a multitude of phase unwrapping algorithms have been proposed over the years, including region-growing algorithms [19], discontinuity minimization algorithms [20], minimum Lp-norm [21] and least-squares [22] algorithms, quality-guided [23,24] and network flow [25] algorithms, branch-cut [26] algorithms and flood-fill [27] algorithms. The processing times of these algorithms range from seconds to minutes, and in some cases hours to generate a standard 640 × 480-pixelphase map [28,29]. Generally, the quality of the phase unwrapping methods increases with theirexecution time [30]. This is due to their iterative nature and local multi-pixel dependency, which inhibits parallel implementation in regular coordinate space. If the produced phase map contains minimal noise and is sufficiently well sampled between consecutive phase jumps, optimized scan-line algorithms may be employed to boost the speed of quality guided algorithms [31]. However, the above-mentioned imaging techniques cannot always guarantee such conditions and applications that aspire real-time visualization of their results often find the phase unwrapping step to be the bottleneck in their image processing schemes [32].

Two state-of-the-art Fourier-based phase unwrapping techniques were described by Schofield and Zhu [33] and by Volkov and Zhu [34]. The algorithms were developed in the field of electron holography and were designed specifically for superior stability against noise and residues present in the wrapped phase map. At their core, unlike the above-mentioned algorithms, they are both single-iteration methods that employ a combination of Fourier techniques to perform phase unwrapping in reciprocal space. As integration is performed in Fourier space, the algorithms are path-independent and show good noise-resistance in real-space.

In this manuscript, we show that optimized serial implementation of this Fourier-based phase unwrapping technique results in processing times which are limited to several tens of milliseconds, depending on the size of the input phase map. As these processing speeds are insufficiently fast for many real-time imaging applications, we propose a breakthrough in speed by optimizing the phase unwrapping algorithm for execution on parallel hardware. We demonstrate that the total processing time can be reduced by over an order of magnitude when compared to serial code by offloading the algorithm to a standard commodity graphics board. This way, a common bottleneck that is currently present in many real-time imaging applications that require high-speed phase unwrapping can be eliminated. Furthermore, we show that its parallel implementation preserves the algorithm's superior performance for noisy images or images with high phase gradients. Additionally, a similar approach will be explored to solve the phase unwrapping problem by representing the 2D wrapped phase map as a set of sparse linear equations and by optimizing the minimization procedure of the preconditioned conjugate gradient (PCG) algorithm in Fourier space. The major advantage of this approach is that it allows the user to weight the iterative PCG solver towards areas of high quality phase data and away from areas within the phase map in which noise is expected to corrupt the phase unwrapping procedure. By providing the algorithm with a local quality map that quantifies pixel validity, convergence of the minimization procedure can be accelerated.

2. Method and Implementation

2.1. Schofield, Volkov and Zhu Phase Unwrapping

The general problem of phase unwrapping boils down to the correct extraction of the true phase $\varphi(\vec{r})$ from the wrapped phase $\varphi_W(\vec{r})$ by constructing an integer number field $k(\vec{r})$ such that (1) becomes an identity for all $\vec{r} \in \Omega_N$:

$$\varphi(\vec{r}) = \varphi_W(\vec{r}) + 2\pi k(\vec{r}) \quad (1)$$

Although both methods developed by Volkov and Zhu and Schofield and Zhu aim to solve (1) using a set of Fourier techniques, their specific implementations differ. Whereas the Volkov algorithm combines the use of several Fourier transform properties with the symmetrization rule [35] to obtain, in one go, the whole field of integer numbers $k(\vec{r})$, the Schofield algorithm formulates (1) in terms of Laplace operators:

$$k(\vec{r}) = \frac{1}{2\pi}\nabla^{-2}[\nabla^2\varphi(\vec{r}) - \nabla^2\varphi_W(\vec{r})] \quad (2)$$

where ∇^2 and ∇^{-2} represent the forward and inverse 2-dimensional Laplace operators, respectively. Next, Equation (2) can be solved using fast Fourier techniques for the Laplacians [36]:

$$\nabla^2 f(x,y) = -\frac{4\pi^2}{MN} IDCT\{(p^2+q^2)DCT[f(x,y)]\} \tag{3}$$

$$\nabla^{-2} g(x,y) = -\frac{MN}{4\pi^2} IDCT\left\{\frac{DCT[g(x,y)]}{(p^2+q^2)}\right\} \tag{4}$$

given an $M \times N$-pixel sized image with (x,y) its real space and (p,q) its Fourier space pixel coordinates. By using discrete cosine transforms (DCT) and inverse discrete cosine transforms (IDCT) instead of general fast Fourier transforms (FFT) [37], the Neumann boundary conditions that require the gradient of $k(\vec{r})$ normal to the original image boundary to vanish is met and the additional memory transfers required by the symmetrization rule can be avoided. As it is our goal to achieve fast execution of the algorithm on parallel hardware, it is best to avoid as many additional memory transfers as possible. Therefore, the design principle of the Schofield algorithm was chosen for further parallel implementation. In order to solve (2) without prior knowledge of the true unwrapped phase $\varphi(\vec{r})$, the authors define the complex quantity $P(r) = exp[i\varphi_W(\vec{r})] = exp[i\varphi(\vec{r}) - i2\pi k(\vec{r})] = exp[i\varphi(\vec{r})]$ and recognize that $Im\big((1/P)\nabla^2 P\big) = \nabla^2\varphi$ [38], where $Im(\ldots)$ indicates the imaginary part. The combination of these identities yields an expression for the Laplacian of the unwrapped phase, using only knowledge of the wrapped phase:

$$\nabla^2\varphi = \cos\varphi_W \nabla^2(\sin\varphi_W) - \sin\varphi_W \nabla^2(\cos\varphi_W) \tag{5}$$

Combining Equations (3–5) along with enforcing the constraint that $k(\vec{r})$ must be an integer number results in a single-step solution for Equation (2) and ultimately in a fast and deterministic solution for Equation (1).

2.2. Preconditioned Conjugate Gradient Phase Unwrapping

The Schofield, Volkov and Zhu phase unwrapping technique implicitly requires correct phase data information to be present across the entire wrapped phase map. In many practical applications, this is not the case. For such applications, input data weighting based on user-defined quality masks may be desirable. To this end, we propose to employ the approximate solution obtained with the Schofield, Volkov and Zhu technique as a first estimator of the unwrapped phase and iteratively weigh the obtained solution using a preconditioned conjugate gradient algorithm. The PCG method of 2D-weighted least-squares phase unwrapping is presented in detail in numerous papers [39–42]. For a thorough treatment of conjugate gradient methods, the reader is referred to Kershaw *et al.* [43]. First, an approximate solution is obtained using Equations (1–5). This solution is then used as a preconditioned input to an iterative solver which converges towards the exact solution in N or less iterations for an $N \times N$ problem. However, in practice, the preconditioning step greatly increases convergence speed so that the problem may be solved sufficiently well in only a few iterations.

In essence, the weighted PCG algorithm requires iterative minimization of:

$$\varepsilon^2 = \sum_{i=0}^{M-2}\sum_{j=0}^{N-2}\left(\varphi_{i+1,j} - \varphi_{i,j} - \Delta_{i,j}^{x}\right)^2 + \sum_{i=0}^{M-2}\sum_{j=0}^{N-2}\left(\varphi_{i+1,j} - \varphi_{i,j} - \Delta_{i,j}^{y}\right)^2 \tag{6}$$

where $\varphi_{i,j}$ is the discrete wrapped phase value at coordinate point (i,j) and $\Delta_{i,j}^{x}$ and $\Delta_{i,j}^{y}$ represent the phase differences of the unwrapped phase grid in the *x*- and *y*-direction, respectively. This minimization problem can be solved iteratively, as shown in the pseudo-code below:

```
for k = 0 → Max Iterations
    Apply preconditioning step (Equations (1 − 5))
    Update solution using weighted conjugate gradient method
    Check for convergence
      if Convergence then
         Exit loop
      end if
end for
```

The weighted conjugate gradient method allows the user to influence the relative importance of different phase values in coordinate space by amplifying or reducing their discrete contribution to the total sum of ε^2.

2.3. Parallel Implementation of the Discrete Cosine Transform

In order to prepare the algorithm for optimized execution on parallel hardware, we write the discrete cosine transform as a combination of matrix multiplications. As the DCT is a separable linear transformation, the two-dimensional variant is equivalent to a one-dimensional DCT performed along the first dimension, followed by a one-dimensional DCT along the second dimension. Hence, the two-dimensional DCT and corresponding IDCT for an $M \times N$ input image A and output image B can be defined as [44]:

$$\begin{aligned} DCT(A)_{p,q} &= B_{p,q} \\ &= \alpha_p \alpha_q \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} A_{m,n} \cos\frac{p\pi(2m+1)}{2M} \cos\frac{q\pi(2n+1)}{2N} \end{aligned} \tag{7}$$

$$\begin{aligned} IDCT(B)_{m,n} &= A_{m,n} \\ &= \sum_{p=0}^{M-1} \sum_{q=0}^{N-1} \alpha_p \alpha_q B_{p,q} \cos\frac{p\pi(2m+1)}{2M} \cos\frac{q\pi(2n+1)}{2N} \end{aligned} \tag{8}$$

with

$$\begin{cases} 0 \le p \le M-1 \\ 0 \le q \le N-1 \end{cases} \text{and} \begin{cases} 0 \le m \le M-1 \\ 0 \le n \le N-1 \end{cases}$$

where the normalization factors α_p and α_q are defined as:

$$\alpha_p = \begin{cases} \frac{1}{\sqrt{M}}, p = 0 \\ \sqrt{\frac{2}{M}}, 1 \le p \le M-1 \end{cases} \tag{9}$$

$$\alpha_q = \begin{cases} \frac{1}{\sqrt{N}}, q = 0 \\ \sqrt{\frac{2}{N}}, 1 \leq q \leq N-1 \end{cases} \quad (10)$$

If A is a square matrix, the 2D-DCT can be computed as:

$$DCT(A) = T * A * T' \quad (11)$$

where T represents the square matrix containing the two-dimensional forward DCT coefficients that can be obtained through orthogonalization of Equation (7) [45]. The "*" and "′" symbols denote matrix multiplication and matrix transposition, respectively. Alternatively, when A is a non-square rectangular matrix containing M rows and N columns, a set of square transformation coefficients F_L (size $M \times M$) and F_R (size $N \times N$) can still be constructed from Equation (7) such that:

$$B = DCT(A) = F_L * A * F_R \quad (12)$$

for any $M, N \neq 0$. Likewise, the couple of transformation matrices I_L (size $M \times M$) and I_R (size $N \times N$) that perform the 2D inverse discrete cosine transform can be constructed from Equation (8):

$$IDCT(B) = I_L * B * I_R \quad (13)$$

For a succession of DCTs on matrices of any given size, the transformation matrices F_L, F_R, I_L and I_R need to be determined only once. This way, the calculation of the DCT or IDCT transform of matrix A can be reduced to two matrix multiplications (MM).

3. Numeric and Experimental Results

3.1. Schofield, Volkov and Zhu Phase Unwrapping

The vast majority of computational demands required by the presented phase unwrapping technique are represented by a total of six (three forward and three inverse) discrete cosine transforms. Recently, implementation optimization of the DCT on both parallel and sequential hardware has been the object of study [46,47]. Considerable speed-ups have been reported by implementing the DCT algorithm on the GPU [48,49], without loss of accuracy [50]. However, in previous work [51], it was concluded that the central processing unit (CPU) of a standard personal computer benefits most from the symmetrization rule and FFT-based DCT implementation, whereas, in contrast, the parallel architecture of the graphics processing unit (GPU) clearly favors the matrix multiplication-based approach. To this end, we have implemented the entire phase unwrapping algorithm using highly optimized CPU-code on the one hand, and in parallel code on the GPU on the other hand, each with its respective DCT implementation (CPU-FFT and GPU-MM) that results in faster execution speeds. Both CPU-and GPU-based algorithms were benchmarked on the same machine, consisting of an i5-660 processor (4MB cache, 3.33 Ghz), an NVidia Geforce GTX 770 graphics processing unit and 16 GB of RAM memory. In order to provide a fair comparison between CPU- and GPU-based versions of the presented algorithm, both were implemented using C++ code that was specifically optimized for sequential and parallel execution, respectively. The FFTW-package [52] was used to boost the calculation speed of the FFT and the Intel SSE2-instruction set was fully exploited to maximize the floating point processing capabilities of the

CPU. The GPU-code was implemented using NVidia's compute unified device architecture (CUDA) and was customized for maximum performance on all 1536 present CUDA processing cores. Furthermore, CUDA's basic linear algebra subroutines (CUBLAS) library was used to optimize the matrix multiplications. Source code was generated and managed in Microsoft Visual Studio and its Visual Profiler with NVidia's Nsight plug-in was used for benchmarking. Executable code can be downloaded freely at the website of the University of Antwerp [53].

The single precision (4 bytes per pixel value) execution times of both CPU-and GPU-based phase unwrapping methods were averaged over 1000 iterations of the algorithm, for a set of varying image pixel sizes. The results are included in Table 1. In addition, processing times of the isolated DCT and IDCTs, including and excluding memory transfers from host to device and back, are included in the last two columns of Table 1.

Table 1. Processing times of central processing unit (CPU)-and graphics processing unit (GPU)-based phase unwrapping algorithms, executed on phase maps of varying pixel sizes. The last two columns include the corresponding processing times of optimized GPU-accelerated discrete cosine transforms (DCTs) and inverse DCTs (IDCTs).

Image Pixel Size	CPU (FFT) (ms)	GPU (MM) incl. mem. trans. (ms)	GPU (MM) excl. mem. trans. (ms)	Acceleration Factor (×)	DCT/IDCT incl. mem trans. (ms)	DCT/IDCT excl. mem trans. (ms)
256 × 256	9.6	1.6	1.2	6.0	0.5	0.1
512 × 512	40.2	4.1	3.2	9.8	1.3	0.3
640 × 480	48.5	4.9	4.0	9.9	1.5	0.5
1024 × 1024	182.9	20.1	16.7	9.1	5.9	2.5
2048 × 2048	735.9	136.4	128.4	5.4	27.9	19.8

The second column of Table 1 represents the execution times of the sequential CPU-version of the entire phase unwrapping algorithm whereas the third and fourth columns indicate the corresponding processing times of the parallel GPU-based algorithm, including and excluding the memory transfers of the input image from the host to the device and back, respectively. The GPU-version of the algorithm outperforms the single-core CPU-version by a factor of 5–10 (including memory transfers). This acceleration factor increases linearly as a function of image pixel size up to input maps around 1024 × 1024 pixels where the SSE2 instruction set allows the CPU to increasingly benefit from the inherent parallelism of the single instruction, multiple data (SIMD) computations as well. Nevertheless, the GPU-over-CPU performance gains remain significant. It could be noted that further optimization of the CPU-code for execution on a multi-core CPU with an increasing number of processing cores could reduce this performance difference. However, this approach would ultimately converge to the parallel model. Furthermore, by offloading the workload of the phase unwrapping procedure to the GPU, the CPU is exempt from these calculations and can continue to coordinate other processes in the digital processing pipeline. If we benchmark the individual matrix multiplication-based DCT/IDCTs, we report processing bandwidths ranging between 131 and 204 Megapixels/s if we include the memory transfers from host to device and back. These correspond well to corresponding bandwidths of high-performance GPU-based DCTs reported in literature (141–155 Megapixels/s [48]).

In addition, the GPU processing times without memory transfers from host to device and back are included in the fourth column of Table 1 to include adequate benchmarks for phase unwrapping when the wrapped phase data is already present on GPU memory and does not need to be transferred back to host memory before visualization. As previously stated, the coefficients that are used to calculate the DCT through matrix multiplication are not included in the memory transfer time, as they need to be transferred only once, before the first iteration of the algorithm. Overall, it can be concluded that the GPU implementation of the algorithm is fast enough for most real-time applications, whereas the CPU implementation allows real-time phase unwrapping only on images of low pixel sizes and does not leave the operator with much remaining time for any additional processing.

As the success of the phase unwrapping procedure relies only on the quality of the phase map and not on the nature of the imaging technique itself, the presented algorithm can be used in a variety of applications. Figure 1 contains two such examples, one from the field of electron holography (first row), the second from magnetic resonance imaging (MRI-second row). Figure 1a contains the electrostatic and magnetostatic phase information of a Cobalt nanowire grown directly onto the grid through the use of a focused ion beam [54] and Figure 1d represents the horizontal susceptibility-weighted phase map of a mouse brain acquired with gradient echo phase-sensitive MRI [55].

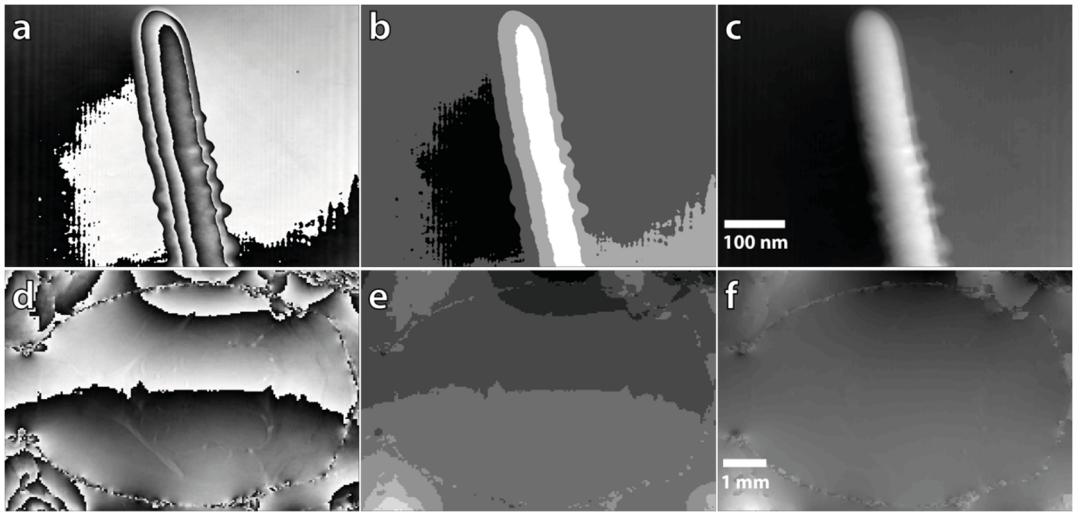

Figure 1. Phase unwrapping procedure for two phase maps obtained with different imaging modalities (first row (**a**–**c**): electron holography, second row (**d**–**f**): phase-sensitive magnetic resonance imaging (MRI)). The first column (a,d) shows the input wrapped phase map; the second column (b,e) depicts the integer number of 2π that is added to the respective wrapped phase map to end up with the unwrapped phase map, included in the third column (c,f). Scale bars are included in the third column of subfigures only but apply to the other images in their respective row, also.

The second column (subfigures b and e) of Figure 1 is included to emphasize the essentially deterministic quality of the presented method. It shows the $k(\vec{r})$-map of integer multiples of 2π, corresponding to the solution of (2) that need to be added to the wrapped phase images of the first column pixel-per-pixel to acquire, finally, the unwrapped phase images (subfigures c and f) of the third column of Figure 1. Although the examples contain high phase gradients, noise and phase residues, the presented

method succeeds in unwrapping them correctly. It should be noted, however, that since the boundary conditions are merely approximated and a rounding step is introduced in the solution for the $k(\vec{r})$-map, the presented Fourier-based algorithm may not completely unwrap the phase map when correct phase information is not present in the entire image. For this class of images, additional iterations of the algorithm, possibly in combination with interpolation of the phase map, may be needed.

3.2. Preconditioned Conjugate Gradient Phase Unwrapping

As the Fourier-based phase unwrapping technique implicitly requires a continuous phase distribution of the original, unwrapped phase, the algorithm is known to fail on images containing regions of invalid phase information. To demonstrate this effect, a random wrapped phase map was generated containing discrete regions where phase data was deliberately set to constant zero. The failure of the algorithm can be seen in the second column of Figure 2. The integer field $k(\vec{r})$ of multiples of 2π (first row) shows discontinuous jumps (circled in red) that do not correlate with the actual phase jumps, leading to incorrectly unwrapped phase maps (second row). However, by providing the PCG algorithm with an input mask of regions where we know the data is invalid (second row, first column), the approximate solution to the single-step Fourier algorithm (second row, second column) can be used as preconditioner (IT1) or first estimator in an iterative weighted conjugate gradient process. In the presented example, the minimization procedure of Equation (6) produces a correctly unwrapped phase map within two additional iterations after initial preconditioning, by weighting the intermediate solutions to the conjugate gradient method with a simple binary pixel map. The second to fourth columns of Figure 2 indicate that, in the presented example, the phase misses are reduced from seven instances (IT1) to one miss (IT2) to none (IT3).

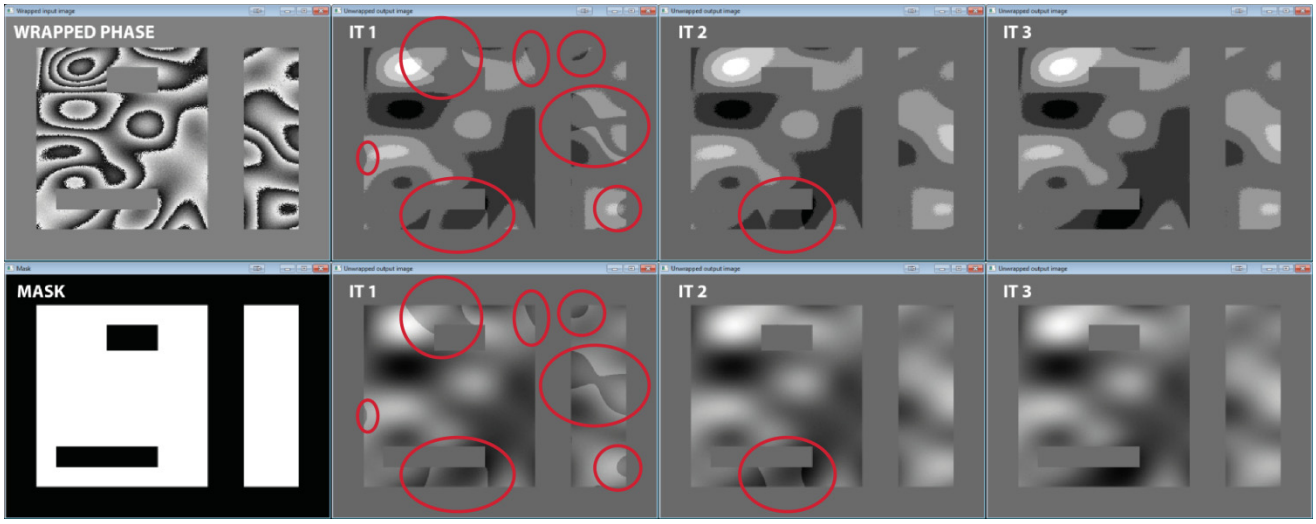

Figure 2. The preconditioned conjugate gradient algorithm is able to correctly unwrap a randomly generated phase map containing regions where phase data is set to zero in three iterations (IT 1-3) by weighting the intermediate preconditioned conjugate gradient (PCG) solutions according to a user-defined input quality mask. Using a simple binary mask, the phase misses (circled in red) are reduced from seven to none in three iterations.

Construction of the preconditioner required 4.9 ms for a phase map containing 640 × 480 pixels (see Table 1), and required an additional 2.1 ms per iteration. Therefore, calculation of the third generation integer field $k(\vec{r})$ in the presented example required a total processing time of 9.1 ms. This shows that the PCG-minimization procedure can be used effectively in combination with Fourier-based phase unwrapping to process input phase maps with discernible regions of invalid or corrupt data whilst still retaining real-time processing speeds. Depending on the prevalence of these regions, the user can specify a number of additional iterations which are to be calculated, at an additional processing time penalty per iteration.

Again, executable code of the preconditioned conjugate gradient-based phase unwrapping algorithm can be downloaded freely at the authors' website [53].

3.3. Real-Time Optical Profilometry

In order to evaluate the presented method qualitatively and to demonstrate its real-time processing capabilities, the phase unwrapping algorithm was implemented in a high-speed optical profilometry setup, similar to the one presented by Huang *et al.* [56]. A set of three phase-shifted line-patterns is projected onto the target at a rate of 60 projections per second. When observed by a camera with adequate frame rate, placed at an angle with the projection direction, 20 phase maps of 640 × 480 data points can be calculated from the recorded images each second. As the resulting phase values are wrapped to the finite interval $[-\pi, \pi]$, phase unwrapping needs to take place before scaling transformations can link them to their respective height values. A single-frame excerpt from such a multi-image recording session (Media 1 [53]) is shown in Figure 3. It can be noticed that the simultaneous phase calculation, phase unwrapping procedure and multiple visualization operations can easily be completed in the provided time window of 50 ms, representing the current limit of the employed projector-camera system.

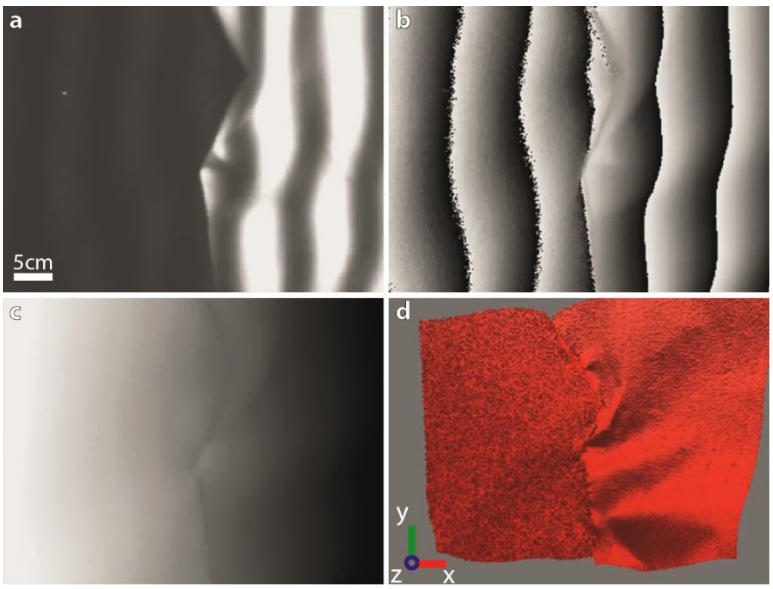

Figure 3. Single-frame excerpt from a multi-image video recording of real-time optical profilometry measurements of moving pieces of fabric (Media 1). (**a**) One of three phase-shifted input images. (**b**) Wrapped phase map. (**c**) Unwrapped phase map. (**d**) 3D perspective-view of (c).

To highlight the robustness of the proposed phase unwrapping algorithm, a combination of highly reflective (right side) and poorly reflective (left side) fabric was chosen as measurement object. The low amount of light reflected from the left side of the object results in a lower local contrast of observed projected line patterns, as can be seen in Figure 3a. This produces a much noisier phase distribution with less fluent phase jumps (Figure 3b), with a shadow effect at the border of both pieces of fabric. Nevertheless, the phase unwrapping algorithm is able to cope with these regions of poor phase quality perfectly as the fabric moves and shifts at reasonably high speeds. The corresponding unwrapped phase map (Figure 3c) shows no phase discontinuities throughout the recording session. An additional 3D perspective-view of (c), including vertex and fragment shading, is included in Figure 3d to allow for better depth perception of the object's height distribution.

4. Conclusions

By combining novel Fourier-based phase unwrapping techniques with a custom parallel implementation of the two-dimensional discrete cosine transform and by optimizing its execution on the graphics processing unit of a standard low-cost graphics card, we have presented a real-time phase unwrapping algorithm that is robust in the presence of noise and phase vortices. Furthermore, its processing time is dependent only on the input pixel size and not on the phase jump density or noise level present in the image. The single-step and deterministic algorithm can unwrap the phase of input images of relatively large pixel sizes at previously unreported speeds, thereby eliminating a bottleneck that is commonly present in many applications in real-time optical profilometry and interferometry, as well as in other image processing applications in which phase map images are generated at high speeds. The parallel implementation of an iterative PCG-solver allows for additional user control over the convergence procedure of the phase unwrapping procedure.

Acknowledgments

The authors wish to thank Vyacheslav V. Volkov for sharing his insights with regard especially to the issue of specific boundary condition implementation, Elsa. Javon and Marleen Verhoye for producing the electron holography and MRI images, respectively.

Author Contributions

Sam Van der Jeught has conducted the research including parallelization and benchmarking of the code. He applied the developed algorithm to various applications and wrote the paper; Jan Sijbers and Joris J. J. Dirckx contributed to the theoretical design and development of the algorithm and aided in the writing of the paper.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Takeda, M.; Mutoh, K. Fourier transform profilometry for the automatic measurement of 3-D object shapes. *Appl. Opt*. **1983**, *22*, 3977–3982.
2. Srinivasan, V.; Liu, H.C.; Halioua, M. Automated phase-measuring profilometry of 3-D diffuse objects. *Appl. Opt.* **1984**, *23*, 3105–3108.
3. Takeda, M. Spatial-carrier fringe-pattern analysis and its applications to precision interferometry and profilometry: An overview. *Ind. Metrol.* **1990**, *1*, 79–99.
4. McDonach, A.; McKelvie, J.; McKenzie, P.; Walker, C.A. Improved moire interferometry and applications in fracture mechanics, residual stress and damaged composites. *Exp. Tech.* **1983**, *7*, 20–24.
5. Dirckx, J.J.J.; Buytaert, J.A.N.; Van der Jeught, S. Implementation of phase-shifting moiré profilometry on a low-cost commercial data projector. *Opt. Laser Eng.* **2010**, *48*, 244–250.
6. Haacke, E.M.; Mittal, S.; Wu, Z.; Neelavalli, J.; Cheng, Y.C. Susceptibility-weighted imaging: Technical aspects and clinical applications, part 1. *Am. J. Neuroradiol.* **2009**, *30*, 19–30.
7. Gatehouse, P.D.; Keegan, J.; Crowe, L.A.; Masood, S.; Mohiaddin, R.H.; Kreitner, K.F.; Firmin, D.N. Applications of phase-contrast flow and velocity imaging in cardiovascular MRI. *Eur. Radiol.* **2005**, *15*, 2172–2184.
8. Colesanti, C.; Ferretti, A.; Prati, C.; Rocca, F. Monitoring landslides and tectonic motion with the permanent scatterers technique. *Eng. Geol.* **2003**, *68*, 3–14.
9. Lu, Z.; Mann, D.; Freymueller, J.T.; Meyer, D.J. Synthetic aperture radar interferometry of Okmok volcano, Alaska: Radar observations. *J. Geophys. Res.* **2000**, *105*, 10791–10806.
10. Rignot, E.; Hallet, B.; Fountain, A. Rock glacier surface motion in Beacon Valley, Antarctica, from synthetic-aperture radar interferometry. *Geophys. Res. Lett.* **2002**, *29*, 1–4.
11. Strozzi, T.; Luckman, A.; Murray, T. Wegmuller, U.; Werner, C.L. Glacier motion estimation using SAR offset-tracking procedures. *Geosci. Remote Sens.* **2002**, *40*, 2384–2391.
12. Harp, G.R.; Saldin, D.K.; Tonner, B.P. Atomic-resolution electron holography in solids with localized sources. *Phys. Rev. Lett.* **1990**, *65*, 1012–1015.
13. Tonomura, A. Applications of electron holography. *Rev. Mod. Phys.* **1987**, *59*, 639–669.
14. Schnars, U.; Jüptner, W. Direct recording of holograms by a CCD target and numerical reconstruction. *Appl. Opt.* **1994**, *33*, 179–181.
15. Schumann, W.; Dubas, M. *Holographic Interferometry: From the Scope of Deformation Analysis of Opaque Bodies*; Springer Series in Optical Sciences; Springer-Verlag: Berlin, Germany; New York, NY, USA, 1979; Volume 16.
16. Itoh, K. Analysis of the phase unwrapping algorithm. *Appl. Opt.* **1982**, *21*, 2470.
17. Huntley, J.M. Noise-immune phase unwrapping algorithm. *Appl. Opt.* **1989**, *28*, 3268–3270.
18. Cusack, R.; Huntley, J.M.; Goldrein, H.T. Improved noise-immune phase-unwrapping algorithm. *Appl. Opt.* **1995**, *34*, 781–789.
19. Xu, W.; Cumming, I. A region-growing algorithm for InSAR phase unwrapping. *Geosci. Remote Sens.* **1999**, *37*, 124–134.
20. Flynn, T.J. Two-dimensional phase unwrapping with minimum weighted discontinuity. *J. Opt. Soc. Am. A* **1997**, *14*, 2692–2701.

21. Ghiglia, D.C.; Romero, L.A. Minimum Lp-norm two-dimensional phase unwrapping. *J. Opt. Soc. Am. A* **1996**, *13*, 1999–2013.
22. Pritt, M.D.; Shipman, J.S. Least-squares two-dimensional phase unwrapping using FFT's. *Geosci. Remote Sens.* **1994**, *32*, 706–708.
23. Herráez, M.A.; Burton, D.R.; Lalor, M.J.; Gdeisat, M.A. Fast two-dimensional phase-unwrapping algorithm based on sorting by reliability following a noncontinuous path. *Appl. Opt.* **2002**, *41*, 7437–7444.
24. Zhong, H.; Tang, J.; Zhang, S.; Chen, M. An improved quality-guided phase-unwrapping algorithm based on priority queue. *Geosci. Remote Sens. Lett.* **2011**, *8*, 364–368.
25. Curtis, C.W.; Zebker, H.A. Network approaches to two-dimensional phase unwrapping: Intractability and two new algorithms. *J. Opt. Soc. Am. A* **2000**, *17*, 401–414.
26. Gutmann, B.; Weber, H. Phase unwrapping with the branch-cut method: Role of phase-field direction. *Appl. Opt.* **2000**, *39*, 4802–4816.
27. Asundi, A.; Wensen, Z. Fast phase-unwrapping algorithm based on a gray-scale mask and flood fill. *Appl. Opt.* **1998**, *37*, 5416–5420.
28. Ghiglia, D.C.; Pritt, M.D. *Two-Dimensional Phase Unwrapping: Theory, Algorithms, and Software*; Wiley: New York, NY, USA, 1998.
29. Zhang, S. Recent progresses on real-time 3D shape measurement using digital fringe projection techniques. *Opt. Lasers Eng.* **2010**, *48*, 149–158.
30. Judge, T.R.; Bryanston-Cross, P.J. A review of phase unwrapping techniques in fringe analysis. *Opt. Laser Eng.* **1994**, *21*, 199–239.
31. Zhang, S.; Li, X.; Yau, S.-T. Multilevel quality-guided phase unwrapping algorithm for real-time three-dimensional shape reconstruction. *Appl. Opt.* **2007**, *46*, 50–57.
32. Pham, H.; Ding, H.; Sobh, N.; Do, M.; Patel, S.; Popescu, G. Off-axis quantitative phase imaging processing using CUDA: Toward real-time applications. *Biomed. Opt. Express* **2011**, *2*, 1781–1793.
33. Schofield, M.A.; Zhu, Y. Fast phase unwrapping algorithm for interferometric applications. *Opt. Lett.* **2003**, *28*, 1194–1196.
34. Volkov, V.V.; Zhu, Y. Deterministic phase unwrapping in the presence of noise. *Opt. Lett.* **2003**, *28*, 2156–2158.
35. Volkov, V.V.; Zhu, Y.; De Graef, M. A new symmetrized solution for phase retrieval using the transport of intensity equation. *Micron* **2002**, *33*, 411–416.
36. Gureyev, T.E.; Nugent, K.A. Phase retrieval with the transport-of-intensity equation. II. Orthogonal series solution for nonuniform illumination. *J. Opt. Soc. Am. A* **1996**, *13*, 1670–1682.
37. Shi, W.; Zhu, Y.; Yao, Y. Discussion about the DCT/FFT phase-unwrapping algorithm for interferometric applications. *Optik* **2010**, *121*, 1443–1449.
38. Hÿtch, M.J.; Snoeck, E.; Kilaas, R. Quantitative measurement of displacement and strain fields from HREM micrographs. *Ultramicroscopy* **1998**, *74*, 131–146.
39. Eisenstat, S.C. Efficient implementation of a class of preconditioned conjugate gradient methods. *SIAM J. Sci. Stat. Comput.* **1981**, *2*, 1–4.
40. Kaufmann, G.H.; Galizzi, G.E.; Ruiz, P.D. Evaluation of a preconditioned conjugate-gradient algorithm for weighted least-squares unwrapping of digital speckle-pattern interferometry phase maps. *Appl. Opt.* **1998**, *37*, 3076–3084.

41. Saad, Y.; Schultz, M.H. Parallel implementations of preconditioned conjugate gradient methods. In *Mathematical and Computational Methods in Seismic Exploration and Reservoir Modeling*; Siam: Philadelphia, PA, USA. 1985; pp. 108–127.
42. Yang, Q.; Vogel, C.R.; Ellerbroek, B.L. Fourier domain preconditioned conjugate gradient algorithm for atmospheric tomography. *Appl. Opt.* **2006**, *45*, 5281–5293.
43. Kershaw, D.S. The incomplete Cholesky—Conjugate gradient method for the iterative solution of systems of linear equations. *J. Comput. Phys.* **1978**, *26*, 43–65.
44. Ghiglia, D.C.; Romero, L.A. Robust two-dimensional weighted and unweighted phase unwrapping that uses fast transforms and iterative methods. *J. Opt. Soc. Am. A* **1994**, *11*, 107–117.
45. Gailly, J.L.; Nelson, M. *The Data Compression Book*; M&T Books: New York, NY, USA. 1995.
46. Aslantas, V.; Ozer, S.; Ozturk, S. Improving the performance of DCT-based fragile watermarking using intelligent optimization algorithms. *Opt. Commun.* **2009**, *282*, 2806–2817.
47. Chen, W.H.; Smith, C.H.; Fralick, S. A fast computational algorithm for the discrete cosine transform. *IEEE Trans. Commun.* **1977**, *25*, 1004–1009.
48. Fang, B.; Shen, G.; Li, S.; Chen, H. Techniques for efficient DCT/IDCT implementation on generic GPU. In Proceedings of the 2005 IEEE International Symposium on Circuits and Systems, ISCAS 2005, Kobe, Japan, 23–26 May 2005; pp. 1126–1129.
49. Ruan, J.; Han, D.D. An efficient implementation of fast DCT using CUDA. *Microelectron. Comput.* **2009**, *8*, 057.
50. Ghetia, S.; Gajjar, N.; Gajjar, R. Implementation of 2-D discrete cosine transform on GPU. *Int. J. Adv. Res. Electr. Electron. Instrum. Eng.* **2013**, *2*, 3024–3030.
51. Ye, P.; Shi, X.; Li, X. *CUDA-Based Implementation of DCT/IDCT on GPU*; University of Delaware: Newark, DE, USA, 2007.
52. Frigo, M. A fast Fourier transform compiler. *ACM Sigplan Not.* **1999**, *34*, 169–180.
53. Van der Jeught, S. Fast Fourier-Based Phase Unwrapping. Available online: https://www.uantwerpen.be/en/rg/bimef/downloads/fourier-based-phase-/ (accessed on 12 June 2015).
54. Javon, E.; Gatel, C.; Masseboeuf, A.; Snoeck, E. Electron holography study of the local magnetic switching process in magnetic tunnel junctions. *J. Appl. Phys.* **2010**, *107*, doi:10.1063/1.3358219.
55. Bakker, C.J.; de Leeuw, H.; Vincken, K.L.; Vonken, E.J.; Hendrikse, J. Phase gradient mapping as an aid in the analysis of object-induced and system-related phase perturbations in MRI. *Phys. Med. Biol.* **2008**, *53*, doi:10.1088/0031-9155/53/18/N02.
56. Huang, P.S.; Zhang, S.; Chiang, F.P. Trapezoidal phase-shifting method for three-dimensional shape measurement. *Opt. Eng.* **2006**, *44*, 123601.