On the Phase Unwrapping of Speckle Interferometry Maps

M.J. Huang, Professor, Mechanical Engineering, National Chung Hsing University, 250, Kuo-Kuang Road, Taichung, Taiwan 40227, R.O.C.

ABSTRACT

This paper aims to provide an effective and distortion-free approach for the phase unwrapping of digital speckle pattern interferometry (DSPI) maps. DSPI is a very powerful measurement tool among industrial users. However, high percentages of noise are present in the experimental data obtained from the method and, therefore, most phase unwrapping algorithms fail to restore the correct phase filed. A common and useful approach to bypass the aforementioned problem is to filter the speckle noise to an acceptable level that phase unwrapping work can forward. However, depending on the filtering algorithm used, more or less degree of distortion is brought into the filtered result. Therefore, another approach to circumvent speckle induced unwrapping problems included branch cut method, discrete cosine transform method, minimum $L^p$ norm formulation, weighted and unweighted least-squares method, and so on. Unlike them, the author proposed a noise-immune algorithm, which could effective circumvent the residues-related problems and enable the unwrapping work of experimental DSPI map. Counting the effective number of pixels with values beyond the assigned threshold around neighborhood of the processed pixel could decide the $2\pi$ ambiguity of the processed pixel correctly and easily. This method is very simple and especially effective for DSPI phase retrieval.

1. Introduction

Phase unwrapping (PU) consists of retrieval of the true phase field from wrapped format data, which is restricted in a $[-\pi, \pi]$ for $2\pi$ modulo. This problem is encountered in several fields like the phase stepping technology [1-4], direct interferometry [5] and the Fourier method [6-8]. Different phase unwrapping algorithms have been proposed from simple algorithm to complex and effective techniques. However, it is usual that reliable unwrapping algorithm can lead to time-consuming processing. Generally, the path-dependent unwrapping algorithm [9] unwraps phases line by line along a predefined path. Any inconsistent point will propagate error along the predefined path to the boundary and cause gaps in the field. Therefore, it is suitable only for phase map without inconsistency or has to be integrated along properly planned path. On the contrary, the path-independent phase unwrapping method restores the data by following all possible paths between any two adjacent points.

In practical situations where high percentage of noise are present in the data set obtained experimentally which might be caused by speckle noise, electronic noise, sampling aliasing and/or the data in nature, thus the unwrapping problems are much difficult to deal with. Digital speckle pattern interferometry (DSPI) [10] was developed in the early 70s as a method of producing interferometric data without using conventional holographic recording techniques. The premise was to use a video camera in stead of holographic plate to record a low spatial frequency fringe pattern. The low resolution of commercial video camera can accept interference fringe only of on-axis condition. In addition, to facilitate the correlation relationships between inspection stages, the scattered speckle patterns should be recorded as clearly as possible. Therefore, high spatial frequency speckle noises are inherently carried on the DSPI correlation fringes. The interferogram is created through the process of correlation (by means of the digital subtraction of two speckle patterns) rather than through wave front interference. The DSPI fringes have similar contrast as the holographic fringes. However, the DSPI image appears much more speckled. Using image processing technique to improve the DSPI image is a usual method employed. For example, the DSPI interferogram may be smoothed with a low pass filter, and a second white light image of the object recorded. The filtered (smoothed) interferogram is then superimposed onto the clean image of the object, thus improving the image detail. Although the image quality is improved, however, certain degree of distortion is brought into the filtered image. Consequently, the phase map obtained from definite frames of filtered phase shifting interferograms is of phase shear mode, which is a troublesome problem for engineers of this field and should be avoided as much as possible. In general, to place cut lines to act as barriers to prevent a further unwrapping algorithm from passing through cut lines is another effective approach [11, 12]. However, for DSPI, its associated phase inconsistencies are extremely dense. The branch cut method is difficult to be applied directly and effectively. Therefore, different kinds of filter [13-17] are proposed to reduce the number of phase inconsistencies to make the branch cut method easier to be applied. Nevertheless, filtering work could distort the phase data of the processed map and cause filtering aliasing.
To bypass the aforementioned problems, we developed new noise immune algorithms [18,19] to restore the DSPI phase map straightly without any pre-filtering work of the speckle interferograms. The proposed noise immune algorithms are classified as a path independent approach and in a parallel manner. They should be categorized as similar approach as the cellular automata phase unwrapping algorithm, which was developed by Ghiglia et al. [20] and was a well-known technique for researches related. However, there had been several serious problems that Ghiglia et al. encountered (e.g., oscillations and large memory demand). To keep track of whether reaches of period-two oscillation, the two most recent iterative maps should be stored in the computer memory for verification, which could greatly limit the unwrapping speed. Although the proposed algorithms are classified as a similar approach as the cellular automata algorithm, the newly developed algorithms are much more concise and robust. In addition, without any global iteration of our present study, the newly developed algorithms enable the correct stitching between segmented submaps, thus can further enhance the unwrapping efficiency of the algorithm, especially for map of great size. Successful unwrapping can be easily achieved without regarding to phase inconsistencies of a DSPI map, whereas for conventional phase unwrapping algorithm (including the cellular automata algorithm) the phase inconsistencies of a map should be properly processed beforehand to avoid any non-conservative phase retrieval of a DSPI wrapped map. Applying the proposed noise immune algorithms to an experimental DSPI wrapped map proves the effectiveness of the proposed algorithms.

2. DSPI Technique
The data produced by a DSPI system is similar to that of a holographic system, in that each fringe represents a line of constant (phase) contour. The main difference between a holographic interferogram and a specklegram is the image quality. Because of the speckle noise associated with DSPI, details of the object’s surface are often obscured. This puts a limit on the density of resolvable DSPI interferometric fringes and affects the maximum displacement that the system can measure. Since a speckle data is created and updated at video rate, this characteristic makes DSPI a perfect technique for in-situ inspection of production parts.

With DSPI, light from distorted sample reaches the recording plane, interferes with the reference beam and creates a slightly modified speckle pattern. The new speckle pattern is then subtracted from the original pattern. The result is a video frame containing the phase changes between inspection stages. The subtracted intensity contains a high frequency speckle noise term, which is the carried wave of the system and a low frequency term, modulated by the phase change signals of interest.

3. Review of current methods (in parallel approaches)
Parallel phase unwrapping is a specific branch for phase retrieval. It is path independent and in iterative approaches. Some current methods are reviewed.

3.1 Cellular Automata Algorithm
The cellular automata (CA) method for phase unwrapping was proposed by Ghiglia et al. [20] in 1987. It offered promise of computation in nondirectional, parallel manner. Basically, any point flagged as inconsistency by the 2 x 2 point path checker is not included in any automata decision-making process. Only the unflagged points are used in the local neighborhood test. The complete 2D algorithm is as follows:

1. Each site looks at all unflagged neighbors within a distance of one unit. The phase differences between the site of interest and each neighbor are computed.
2. The strength of votes is accumulated.
3. The site is changed in value by 2π rad in a direction appropriate to the accumulated strength-of-vote.
4. If none of the neighbors differ by more than π rad from the current site, no change is made to the site value.
5. Repeat steps 1 through 4 until the period-two oscillatory state is reached, then average two oscillatory states. If the phase is not unwrapped, go to step 1; otherwise terminate.

Through local (steps 1~4) and global (step 5) iterations, the cellular automata algorithm can successfully restore correct phase field of map with discrete monopoles. For ESPI phase map, the phase inconsistencies are densely distributed and are mutually coupled between one another. This causes failure of the cellular automata method.

3.2 Parallel Noise-Immune Algorithm
The unwrapping algorithm [19] is a parallel path independent process of removing phase jumps by local neighborhood tests and phase retrivals. The algorithm is that any phase jumps are parallel detected and removed in a pre-selected sense (+2π or −2π). The processed map is further implemented by replication of a series of above operations that successively better approximates the result to produce a continuous phase field. The implementation could be performed in two alternatives: upward (+2π) or downward (−2π); either mode could work without any limitation difference. Taking upward unwrapping as an example, 2π phases are parallel added to the detected pixels by local neighborhood tests to remove the phase jumps. The iteration terminates whenever the phase jumps are totally eliminated. The unwrapping processes are summarized as below:

1. Choose one unwrapping criterion (low-to-high or high-to-low) as the rule.
2. Parallel phase jump verification is implemented to every pixels of the map by neighboring four (or eight) rules.
(3) Based on the predefined unwrapping direction, 2\(\pi\) phase is added (or subtracted) to the data of the pixels whose phase level is \(\pi\) lesser (or greater) than any of its neighbors.

(4) While the entire phase jumps are eliminated, the iteration terminates. Otherwise, go to step 2 for next iteration.

The algorithm described above could only process map without any phase inconsistency. In practical situation (especially for the experimental ESPI wrapped map), severe phase inconsistencies are accompanied. Therefore, the inconsistency-free criterion mentioned above should be further modified to neglect any ill suggestion from phase inconsistent discontinuity and adopt only the well suggestion from phase consistent discontinuity (or jump). Asking more neighbors to vote for a discontinuity is a reasonable and effective way of approach. All votes for a well data will be totally consistent from all voters and, on the contrary, conflict among them for an ill one. Only opinions from influential neighbors are taken into considerations. Moreover, one more redundant is added to circumvent occasional cases. Mathematical expression can refer to the original paper of Ref. [19]. With the integration of the noise-immune criterion, easy work should be done to restore an experimental DSPI map.

### 3.3 Region-referenced Algorithm

For DSPI application, inconsistencies induce serious problems of phase unwrapping. The jump detection based on region-referenced criterion [18] could easily bypass the prohibition of any branch-cut-crossing integration. The region-referenced-based detection criterion is implemented by the comparison between data of the processed pixel and the pixels of its referenced neighbor (instead of the neighboring-points detection technique that CA algorithm used).

Different from the Goldstein's approach [21], which restored the continuous phase field by the (path-dependent) reconstruction without any crossing of the branch cuts, Ref. [18] utilized the associated neighborhood of each pixel to identify whether it is a phase discontinuity or not. Only "good" phase jumps are treated by 2\(\pi\) phase retrieval leaving those "bad" phase jumps remanded unchanged. The detection criterion of the modified version is also governed by the phase differences between data of the processed pixel and that of the referenced pixels. However, instead of being referred to the four (or eight) pixels just near to the processed pixel \((i,j)\), an ensemble of pixels within longer distance are used to provide the data for reference.

The average of a neighbored ensemble is calculated to provide as a referenced level for jump verification. The detection speed will however be slowed down by the averaging implementation. Therefore, an alternative implementation by counting the number of jumps of each referenced set is their approach. Take a 5 x 5 window as an example. For the central pixel, the phase differences with respective to each of the 24 referenced pixels are calculated. The calculated results could be any data within \((-2\pi, 2\pi)\) and are classified into three ranges: \((-2\pi, -\pi]\), \((-\pi, \pi]\), and \([\pi, 2\pi)\). No phase jump is present if the results are within the interval of \((-\pi, \pi]\), and a negative or a positive jump is present if the result is in the interval of \((-2\pi, -\pi]\) and \([\pi, 2\pi)\) respectively. As mentioned above, only one unwrapping direction (\(+2\pi\) or \(-2\pi\)) is applied in our algorithm. Thus, if to apply \(+2\pi\) PU algorithm, only phase differences within \((-2\pi, -\pi]\) are concerned. Contrarily, only phase differences within \([\pi, 2\pi)\) should be detected if \(-2\pi\) PU algorithm is the selected direction of phase retrieval. The total number of flags of each referenced set is further counted and, if the number greater than one half of the number of pixels within the referenced set, that means a jump is present between the central pixel and its referenced neighbor.

The "fringe-shifting criterion" is combined with the "region-referenced detection" technique to successfully circumvent the phase inconsistency related problems. Five kinds of window (neighboring-region, \(1\times1\) to \(8\times8\), shown as Fig. 1(a) ~ (e)) were utilized for verification. Referred to the figure, \((i,j)\) is the processed pixel and, for each case, the gray pixels ensemble is one of the four symmetric neighboring-regions of \((i,j)\). Some important factors with regard to the decision of the referenced mask shape are listed:

1. At least one (among the four referenced masks) should clearly represent the nearby jump ensemble for each pixel within the map.
2. For data near the edges and the corners, the mask shape should be as close as possible to that of pixels valid.
3. The weighting of the phase differences of different pixels within a mask should be as similar as possible.

To make sure that the jump detection suggested by certain kind of window is valid, the threshold was set at a value, which was a little greater than one half of the number of pixels of the window. The suggested threshold of each is listed in the 3rd column of Table 1. It was also known that around the border of the map there are less pixel for reference, therefore, the error was much easier to occur. The design of mask \(\odot\) could successfully circumvent this problem, with its performance superior to that of mask \(\circ\). Therefore, after considering the balance between accuracy and speed, we suggest the window \(\odot\) as the optimum selection among them (i.e., \(\odot\) to \(\ominus\)).

Similarly, the region-referenced concept can be adopted in different means, that is to refer to a group of pixels nearby and to check whether the total numbers of jump exceed a certain threshold or not. This detection rule is much stable than the previous one (neighboring-4 rule) under severe noise cases. Two windows, one a rectangular and the other an octagon, shown as Fig. 2(a) and 2(b), are used to define "neighbors" of a processed pixel. For each of them, the numbers of neighbors are 24 and 36 respectively. To ensure the jump detection is effective, the thresholds are set as one third of each, i.e., 8 and 12 respectively.
Fig. 1. Five kinds of windows used for phase jump detection. (a), (b), (c), (d) and (e) representing I, II, III, IV and V respectively.

Fig. 2. The rectangular (a) and the octagon (b) referenced neighborhoods.
4. Description of the algorithm

Adaptive thresholds are further utilized herein to optimize the performance of phase unwrapping. The mathematical expressions of the proposed algorithm are listed as follows.

4.1 $+2\pi \ PU$

$$[\Phi(i, j)]_{k} = [\Phi(i, j)]_{k-1} + 2\pi \left[h(i, j)\right]_{k},$$

(1a)

where

$$\left[h(i, j)\right]_{k} = \begin{cases} 1, & \sum \sum \left[F(i, j; r, s)\right]_{k} \geq T_{1}, \\ 0, & \text{otherwise} \end{cases}$$

(1b)

and

$$\left[F(i, j; r, s)\right]_{k} = \begin{cases} 1, & \text{if } \left[\Phi(r, s)\right]_{k-1} - \left[\Phi(i, j)\right]_{k-1} \geq T_{2}, \\ 0, & \text{otherwise} \end{cases}$$

(1c)

where \( k \) is the iterative number, \((i,j)\) are the indexes of the pixel of interest, and its phase value is denoted by \( \psi(i,j) \). For a rectangular mask of size \( m \times n \) and with pixel \((i,j)\) as its central pixel, its pixel’s indexes \( r \) and \( s \) run from \( i - (m-1)/2 \) to \( i + (m-1)/2 \) and \( j - (n-1)/2 \) to \( j + (n-1)/2 \), respectively. As the decision criterion of the unwrapping algorithm of Ref. [18], whether \( 2\pi \) phase bias is to be added to phase value of the processed pixel \((i,j)\) or not is decided by the total number of “flags” of the referenced window. Whenever \( \psi(r,s) \) is greater than \( \psi(i,j) \) by a threshold value of \( T_{2} \), a flag is set on pixel \((r,s)\) of the associated window of pixel \((i,j)\). In Eq. (1b), \( F(i,j; r, s) \) stands for the flag value between pixels \((r,s)\) and \((i,j)\). This (flag) value is set as one or null according to the phase differences between them (see Eq. (1c)). In this study, two threshold values should be set to optimize the adaptive performance of the unwrapping criterion. \( T_{1} \) is a threshold for number of pixel, which should be less than the value of \( m \times n \) (the size of referenced window, respectively). \( T_{2} \) is the adaptive threshold for phase differences between two pixels and generally, set as a fraction value between \( \pi \sim 2\pi \).

4.2 $-2\pi \ PU$

$$[\Phi(i, j)]_{k} = [\Phi(i, j)]_{k-1} - 2\pi \left[h(i, j)\right]_{k},$$

(2a)

where

$$\left[h(i, j)\right]_{k} = \begin{cases} 1, & \sum \sum \left[F(i, j; r, s)\right]_{k} \geq T_{1}, \\ 0, & \text{otherwise} \end{cases}$$

(2b)

and

$$\left[F(i, j; r, s)\right]_{k} = \begin{cases} 1, & \text{if } \left[\Phi(r, s)\right]_{k-1} - \left[\Phi(i, j)\right]_{k-1} \leq -T_{2}, \\ 0, & \text{otherwise} \end{cases}$$

(2c)

As the notations of Eq (1), where \( i, j, k, r, s, m, n \) are integers, and for pixels \((r,s)\) of a rectangular window with \((i,j)\) as its center, its indexes run respectively as follows.

$$r = i + g \quad , \quad g = -(m-1)/2 \sim (m-1)/2$$

$$s = j + h \quad , \quad h = -(n-1)/2 \sim (n-1)/2$$

(3)

In either of the above unwrapping criterions, maps with severe residues can be successively unwrapped without any extra treatments.

5. Cut and patch implementation

Though the discontinuity detection criterion could effectively circumvent the phase inconsistent problem, however, the cycles of iteration strongly depended on the size of the map. Therefore, a technique based on a cut and patch criterion [18] has further proposed to reduce the size of the processed map and successfully enhanced the processing efficiency.

The map can be segmented into several sub-maps, processed individually and properly stitched together finally [22], provided that the performances are not greatly affected. However, for a certain map with phase inconsistency, the jump detection for data near the border is much unstable than for data elsewhere. Therefore, the segmentation-and-stitching technique is much easier to cause error on the created boundaries of each sub-map than the technique straightly processing the wrapped map itself. To circumvent this problem and take the advantages of sub-map processing, a newly developed cut-and-patch implementation is utilized to take the work.
The skill is a modified implementation of the previous. Similarly, the wrapped map is (virtually) divided into several sub-maps, e.g., \( u \times v \) each, and is processed sequentially from left to right and top to bottom. For each sub-map, instead of processing the division itself, a bigger one with an extra width is actually processed, i.e., cut from and, after unwrapping, patched back to the map with the extra width serving as the overlapped area for further usage. More specifically, for an \( M \times N \) wrapped map, it is (virtually) divided into \( M/u \) by \( N/v \) sub-maps with \( u \) and \( v \) standing, respectively, for the horizontal and vertical size of each. Take certain division as an example, e.g., the one with corners ABCD (shown as Fig. 3), for this virtual block, a rectangular sub-map (\( AB'C'D' \) in this case) is cut, unwrapped and patched back to the map from which it is segmented. The cut-and-patch implementation runs from left to right and top to bottom successively till the completion of the last (right-bottom) sub-map. It was shown that the unwrapping was performed block by block with an overlapped area, which had been properly processed in the previous iteration and provided as a well conditioned data for further reference.

An experimental wrapped map of 640 x 480 samples was used to verify the performance enhancement of the cut-and-patch implementation. The wrapped map was virtually divided into several sub-maps (e.g., \( L \times L \) each) and the width of overlap was set as 5 pixels. Therefore, each time a rectangular of size \( (L+5) \times (L+5) \) was cut, unwrapped, and then patched back to the map, from which it had been cut previously. The implementation, one after another, sequentially from left to right and top to bottom, converted the map into a piecewise continuous format. Thus, the stitches between sub-maps with different levels are further necessary. In stead of patching directly, the level difference (round to the nearest multiplier of \( 2\pi \)) of the overlap before and after unwrapping is found and applied to the patch to ensure jump-free of the result.

However, null adjustment of the patch is necessary for most practical cases (included the case presented here). The results depict that the cut and patch technique could greatly reduce the processed time needed, especially for the map of grand scale, with the result as satisfactory as well. The processes of case 40 x 40 is illustrated as Fig. 4 with (a)~(d) standing for the 5th, 35th, 94th, and 171th patching map respectively. It is shown that the technique could achieve excellent unwrapping result within an acceptable time.
6. Experiment

The proposed algorithm is applied to an experimental electronic speckle pattern shearing interferometry (ESPSI) with the shearing mechanism composed of a beam splitter and two reflected mirrors. The practical shear is controlled by the relative tilting angle of two reflective mirrors. A force loads on the center of a vertical-edges-clamped aluminum plate (of size 10cm by 10cm). A PZT-driven shifting mirror is used to perform phase stepping. After phase shifting calculations, phase map of Fig. 5a can be achieved. Since referenced window of size 5 x 5 is used, the unwrapped result with $T_1$ and $T_2$ set as 6 and $1.4\pi$, respectively, are shown in Fig. 5b and Fig. 5c shows its 3D plot.

![Fig. 4](image)

**Fig. 4.** The demonstration of the cut and patch implementation. (a), (b), (c) and (d) Representing respectively the 5th, 35th, 94th and 171th iterative result.

![Fig. 5](image)

**Fig. 5.** An experimental work of ESPSI. (a) The phase map obtained from subtraction of phase map of speckle wrapped field after and before deformation. (b) The unwrapped phase field ($T_1 = 6, T_2 = 1.4\pi$). (c) The 3D surface plot.
7. Conclusions
An adaptive noise-immune phase unwrapping algorithm based on a parallel and path-independent processing manner has been successfully developed. With the adjustment of two adaptive thresholds, the unwrapped results are superior to those of Refs. [18], [19] and [20]. In addition, maps with physical shear are much easier to be unwrapped by the newly-developed algorithm provided that the two thresholds are suitably set. Generally, as the processed maps are with more either residues or shears, $T_2$ should be set higher to circumvent the problems related. While it happens, the other threshold $T_1$ should be set lower mutually and suitably. Figure 6 presents the unwrapped results of a simulated noisy wrapped map with 0.64 cycle p-v white noise by the adaptive algorithm. It is shown that the valid thresholds consist of a band crossing from the right upper corner to the left bottom corner in Fig. 6. The notations D and ▲ stand for diverge and iteration won’t terminate, respectively, and the numerical number within circle stands for the number of pixels that were wrongly unwrapped.

![Diagram of Fig. 6](image)

Fig. 6. Optimization of the adaptive thresholds.

8. Acknowledgments
The author would like to thank the National Science Council of the Republic of China for financially supporting this research under contract Numbers NSC92-2212-E-005-021.

9. References