Phase-unwrapping algorithm for noisy phase-map processing

J. A. Quiroga and E. Bernabeu

Automated fringe-pattern processing is important in a great number of industrial applications, such as optical data testing and quality control. One of the main problems that arises with these processes is the automated phase unwrapping of the phase map associated with the fringe pattern. Usually the phase map presents problems such as noise, and low-modulation areas. A new phase-unwrapping algorithm with high noise immunity is presented. The algorithm is easily implemented and can process arbitrary shapes. The main features of this algorithm are the use of a queue for the processing of arbitrary shapes and a selection criterion that determines which pixels are going to be processed.

Key words: Phase unwrapping, automated interferometry, fringe-pattern processing.

1. Introduction

The most usual techniques for the calculation of the phase $\phi(x, y)$ associated with a fringe pattern imply the shifting of a reference phase through known increments (phase-sampling interferometry) or by addition of a substantial tilt to the wave front that produces carrier fringes and then by a further Fourier transformation of the fringe pattern.[1] In all these cases the phase is calculated by means of an inverse trigonometric function (arctan or arccos). All these functions return only principal values, i.e., values in the $[-\pi, \pi]$ interval, generating a discontinuous phase map wrapped into a $[-\pi, \pi]$ interval. For sampled functions the phase jumps tend to π as the sampling frequency reaches the Nyquist frequency, and they tend to 2π as the sampling frequency rises compared with the Nyquist frequency. Furthermore, for noise-free fringe patterns it is enough to seek jumps greater than π and to correct them by addition or subtraction of a 2π offset until the difference between collateral pixels is less than π. These clean fringe patterns are not the usual case; usually they are affected by several error sources, which one must keep in mind to design a phase unwrapping algorithm.

The authors are with the Departamento de Óptica, Universidad Complutense de Madrid, Facultad de Ciencias Físicas, Ciudad Universitaria s/n, Madrid 28040, Spain.

Received 13 August 1993; revised manuscript received 19 April 1994.

0003-6935/94/296725-07$06.00/0.

The error sources that most frequently arise in a fringe pattern are as follows:

(a) Noise, electronic or speckle. The electronic noise is produced during the acquisition of the image. Speckle is due to the reflection of a coherent light beam in rough surfaces.

(b) Low-modulation points that are due to areas of low visibility. The low-modulation points appear as fluctuations in the phase module 2π, which might introduce errors in the phase-unwrapping process.

(c) Abrupt phase changes that are due to object discontinuities. This type of error can lead to logical inconsistencies in the processed phase map; i.e., the phase difference between two points is path dependent.

(d) Violation of the sampling theorem. The fringe pattern must be sampled correctly for the recovering of all the information from the phase module 2π. Thus there must be at least two sampling points per fringe. This imposes a maximum spatial variation of the wave front of π rad per sampling point on the detector. In phase-sampling interferometry the sampling condition is more restrictive; at least three sampling points per fringe are needed.[1]

The algorithm presented in this paper can handle in an automatic way the first two problems listed above [(a) and (b)]. The fourth problem, (d), can be overcome with very little change in the algorithm with the technique described in Ref. 2. To solve the third problem, (c), with this technique, one needs to have *a priori* knowledge of the object or to use the techniques described in Refs. 3 and 4.

2. Current Algorithms

Many algorithms for automated processing of noisy fringe patterns have been proposed. Some of them[5–7] make use of local techniques for the determination of invalid pixels in the phase map. Others[3,4] determine the incorrect areas by means of global techniques.

To detect the invalid data points, Ghiglia *et al.*[7] and Huntley[5] checked for the number of 2π discontinuities in a closed loop around each square of 4 pixels in the phase map. That is, for every point in the image the value

$$\begin{aligned} s(x, y) = &[\phi(x + 1, y) - \phi(x, y)] \\ &+ [\phi(x + 1, y + 1) - \phi(x + 1, y)] \\ &+ [\phi(x, y + 1) - \phi(x + 1, y + 1)] \\ &+ [\phi(x, y) - \phi(x, y + 1)] \end{aligned} \tag{1}$$

is computed, where the brackets denote the modulo 2π operation.

From the above definition, $s(x, y) = 0$ or ± 1. The points with $s(x, y) = \pm 1$ are discontinuity sources. Following Huntley,[5] to make a path-independent phase unwrapping, one needs to place cut lines between invalid points of opposite sign. In an isolated invalid point is close to the boundary, the cut line is placed between the point and the boundary. Recently Huntley *et al.*[8] have reported a new algorithm to place these cut lines in an optimal way.[9]

The calculation of $s(x, y)$ detects only local logical inconsistencies and cannot detect invalid data points, that are due to low modulation or to electronic noise. If these invalid pixels are processed, errors might appear in the unwrapped phase that could propagate far away from the invalid pixel. Bone[6] solves this problem by checking the second differences of the locally unwrapped phase. A limit for the spatial variation of the second differences is imposed. Every pixel is checked, and the points that exceed the limit for the spatial variation for the second difference are masked. This technique masks both the logical inconsistencies and the invalid pixels that are due to low modulation or noise.

Concerning the global methods, Towers *et al.*[4] process the phase map dividing the image in little areas or tiles. Every tile is processed independently from the others. Some of the tiles are labeled as invalid and neglected. The remaining tiles are assembled with a minimum spanning tree. Andrä *et al.*[3] make a preprocessing of the phase map by means of topological techniques. From the preprocessed image the invalid areas are determined. Once the invalid areas are masked, the fringe pattern is processed. These global techniques have a difficult implementation and share the inability to detect pointlike errors in the phase-map image.

3. Description of the Algorithm

The algorithm is based in the following three points:

(1) The analysis of phase unwrapping of Ghiglia *et al.*[7] and Itoh.[9]

(2) A fill-flood technique, for the processing of irregular shapes, described in Vrooman and Maas.[10]

(3) A selection criterion based on the local gradient modulo 2π that drives the direction of processing and decides which points should be invalidated.

A. Analysis of Phase Unwrapping

For the analysis of the phase-unwrapping algorithm we follow the work of Ghiglia *et al.*[7] and Itoh.[9] Therefore an in-depth analysis can be found in the references.

In a one-dimensional case, if we denote $\phi(n)$ as the wrapped phase at point n, $\Phi(n)$ as the continuous phase at point n, W as the wrapping operator, that is, $W[\Phi(n)] = \phi(n)$, and Δ as a differencing operator, that is, $\Delta[\phi(n)] = \phi(n) - \phi(n - 1)$, then the continuous phase at point M can be computed as[7,9]

$$\Phi(M) = \phi(0) + \sum_{n=1}^{M} W\{\Delta[\phi(n)]\}. \tag{2}$$

Applying Eq. (2), we can unwrap the continuous phase by integrating the wrapped differences of the wrapped phase. If the sampling theorem is not satisfied, the continuous phase Φ cannot be recovered without *a priori* knowledge. For example, if we suppose that the second derivative of the phase is smooth, it is possible to use the application of the technique presented by Greivenkamp in Ref. 2.

The extension to a two-dimensional case is straightforward. If Δ_m is the vertical differencing operator and Δ_n its horizontal counterpart, we can recover the continuous phase at point (m, n) by

$$\Phi(m, n) = \Phi(0, n) + \sum_{j=1}^{m} W_1\{\Delta_m[\phi(j, n)]\}, \tag{3}$$

$$\Delta(m, n) = \Phi(m, 0) + \sum_{l=1}^{n} W_2\{\Delta_n[\phi(m, l)]\}. \tag{4}$$

Applying Eq. (3), we can unwrap a two-dimensional phase map by unwrapping the first column, with the result $\Phi(0, n)$, and afterwards continue with all the rows starting in this column. If we want to use Eq. (4) instead of Eq. (3), we must proceed with the first row, and from it continue column by column.

The results shown above present two possible schemes for the unwrapping of a phase map. We can use one of Eqs. (3) or (4) to scan the fringe pattern row by row or column by column. Or, on the other hand, we can use one-dimensional Eq. (2), following a path that goes through all the points in the image without passing by the same point twice. The first technique has difficulty in its implementation when irregular-shape processing is needed. To apply the second technique, we need an algorithm able to fill and to process an irregular shape. This algorithm is discussed in Subsection 3.B.

B. Processing of Irregular Shapes and Phase Unwrapping

With the purpose of processing irregular shapes we followed the fill-flood algorithm described in Ref. 10. The processing part of the algorithm was changed to include a noise-handling procedure. The flow chart of the algorithm is shown in Fig. 1.

A binary mask defines the region of interest. It can be user supplied or it can be defined automatically; for example, by means of the fringe-modulation function.[11]

To process the region of interest, we used a queue-based algorithm.[10] The unwrapping procedure is as follows. For each point in the region of interest the phase is unwrapped in respect to a special point, called the integrator pixel. Any point, once processed, can be the integrator pixel. The integrator pixel plays a special role because the phase at the current position is integrated modulo 2π with respect to the value of the continuous phase at the integrator-pixel position. That is, if P_1 is the position of the integrator pixel, P_2 is the position of the current pixel to be processed, $\Phi(P_1)$ the value of the continuous phase at P_1, and $\phi(P_1)$ and $\phi(P_2)$ are the values for the phase map at points P_1 and P_2, the value of the continuous phase at point P_2 is computed by

$$\Phi(P_2) = \Phi(P_1) + W[\phi(P_2) - \phi(P_1)], \tag{5}$$

where W denotes the wrapping operator. This operation is performed for each valid point in the image. The local gradient modulo 2π information is used to validate each point and to drive the phase-map processing. This is doen by the following steps:

(1) The starting pixel is user selected. The pixel is checked to determine if it is part of the processing mask. If this is not the case, the 3×3 neighborhood is searched for a nonmasked pixel. After that, the starting point is stored as an integrator pixel.

(2) The nonmasked and nonprocessed N-connected neighbors of the integrator are sorted by their gradient modulo 2π with respect to the integrator. The M pixels with higher gradients are neglected, and the remaining neighbors are stored in the queue, keeping the gradient ordering. N represents any standard set of neighbors, and M can be any value, as

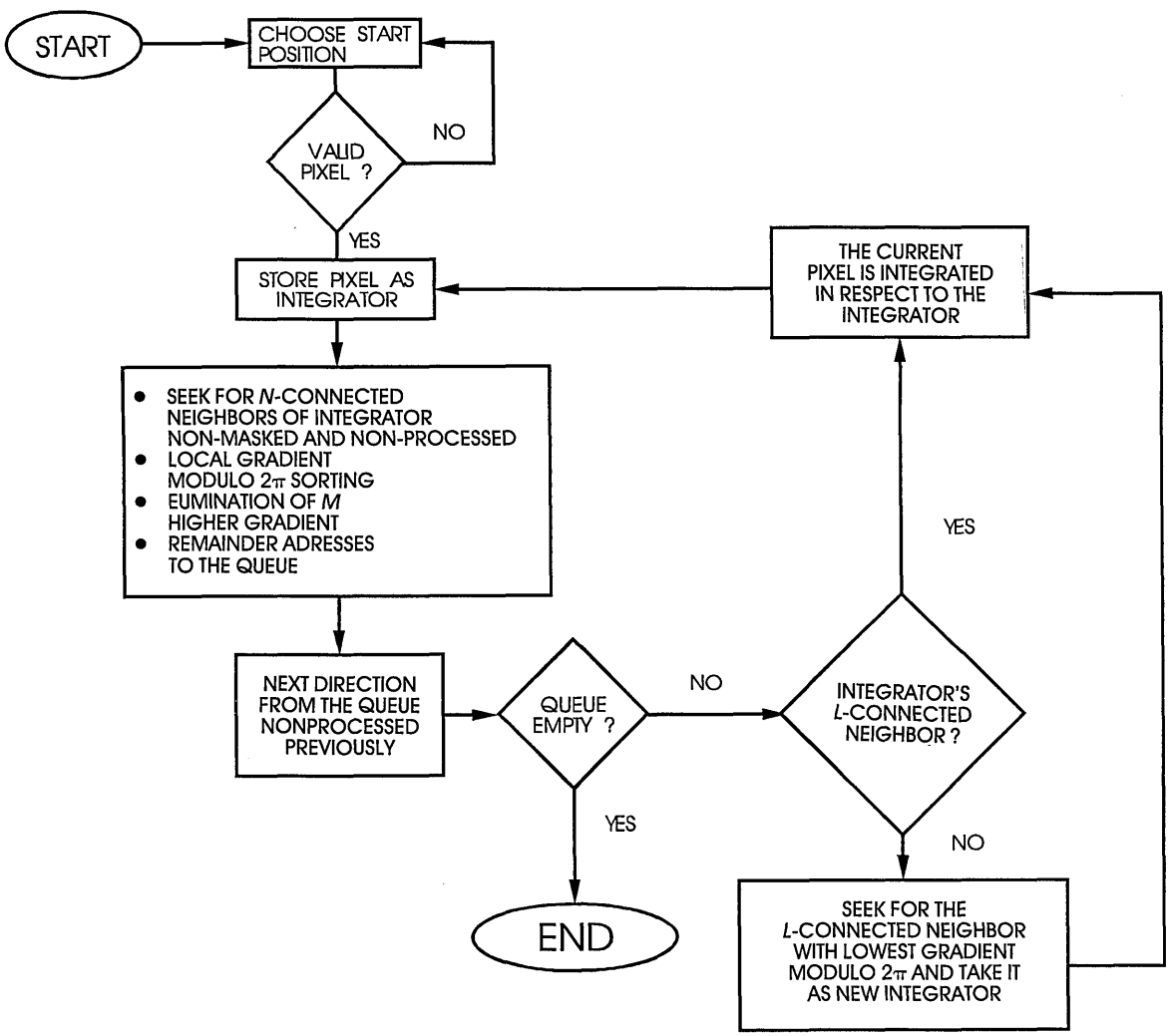


Fig. 1. Flow chart of the algorithm.

it permits the complete processing of the image by the algorithm. We observed, for example, that for noisy phase maps, $M = 5$ is the maximum value for $N = 8$.

(3) The first available pixel from the queue is extracted. If it has been previously processed, the next pixel of the queue is taken until a nonprocessed point is found or the queue is empty. If the queue is empty, the algorithm continues in step (7).

(4) If the integrator and the current pixel are not L-connected neighbors, the processed L-connected neighbor with the lowest gradient modulo 2π with respect to the current pixel is selected as the new integrator. As does N, L represents any standard set of neighbors.

(5) The current pixel is integrated modulo 2π with respect to the integrator by use of Eq. (5).

(6) Return to step (2).

(7) The algorithm ends.

C. Sorting and Exclusion Procedure

The order in which the pixels are stored in the queue is of great importance. The ordering of the pixels in the queue determines locally in which order they are going to be processed. If we were able to find a sorting criterion that could decide which pixels have the higher probability to be invalid ones, we could make them the last ones locally processed. If the invalid (with high probability) pixels are the last ones locally processed, the propagation of the errors associated with them would be stopped or at least limited.

As we explained in Subsection 3.B, the chosen sorting criterion was the local gradient modulo 2π. Every time that the nonprocessed neighbors of an integrator pixel are sought, they are sorted on the basis of their local gradient modulo 2π. Once they are sorted, they are stored in the queue, keeping their sorting. Even with this technique, when the noise level is high, we observed that the errors are propagated, as we show in Section 4. For this reason a modified selection rule was necessary. The modified criterion is based in the redundant structure of the algorithm presented in Subsection 3.B. The term redundant refers to the number of times a pixel appears in a searching operation. The change performed in the algorithm consists of the systematic rejection of the M pixels with higher gradient modulo 2π that appear in every searching operation. This implies that, if there is a pixel that at all times has its gradient modulo 2π among the M highest ones, it will be considered an invalid pixel and will not be processed at all. A high M value will make the algorithm safer with respect to the noise; on the other hand, if the value of M gets high enough that $N - M < 3$, the algorithm can be unusable.

4. Results

To test the algorithm, we used a computer-generated noise-corrupted phase map and a noisy speckle interferogram.

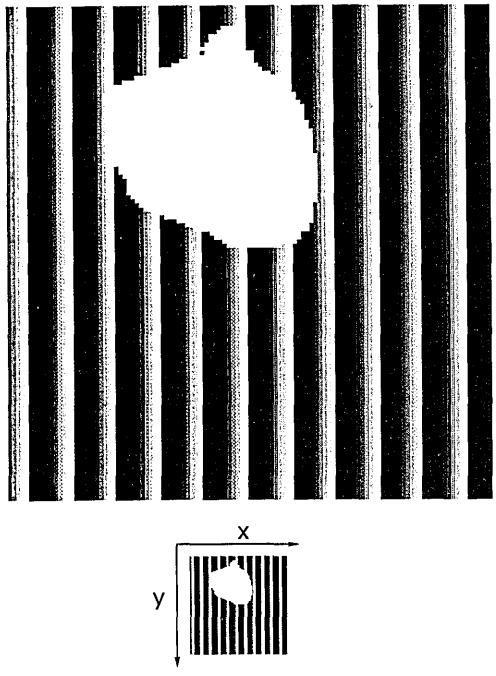


Fig. 2. Phase map with processing mask, representing a positive ramp from left to right. The orientations of the X and Y axes are also shown. This orientation is the same for Figs. 3–5.

The computer-generated phase map corresponds to a positive ramp from left to right. The clean phase map is shown in Fig. 2. To create the corrupted phase map, we computer generated a fringe pattern. The fringe pattern was digitized to 121×128 points with eight bits per pixel; that is, with 256 possible gray levels per pixel. The noise added to the fringe pattern was white, uniformly distributed from -127 to 127 gray levels, and smoothed by a 3×3 pixel averaging window. The corrupted phase map computed from the noisy fringe pattern is shown in Fig. 3.

In order to show the importance of the sorting by local gradient modulo 2π and by further rejection of the higher local gradients, we used three variations of

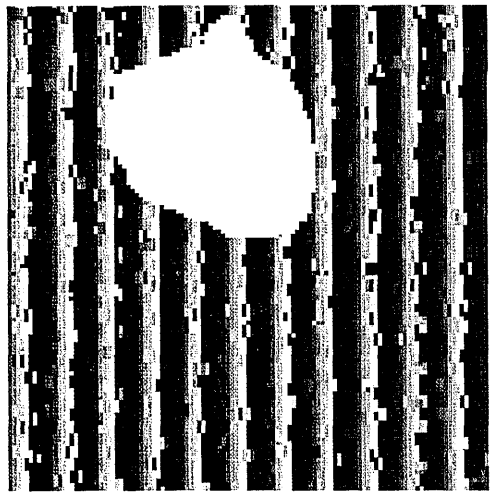
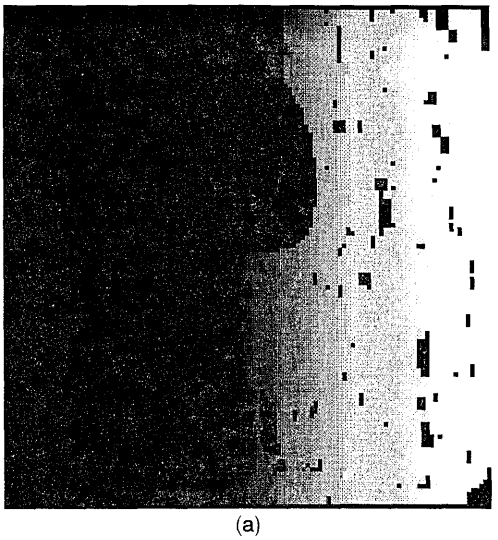
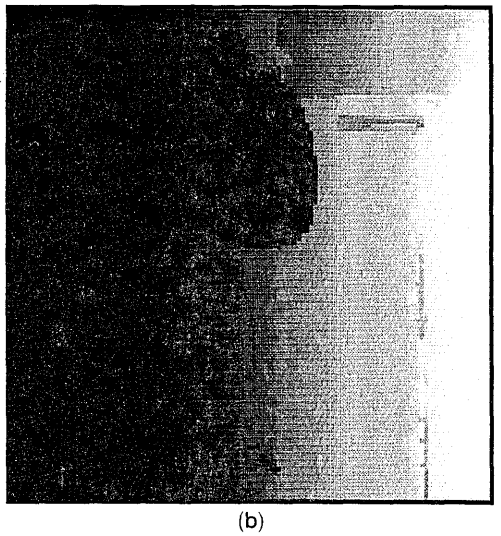


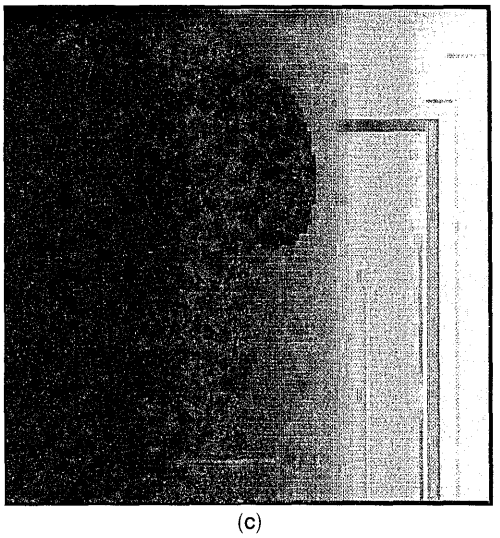
Fig. 3. Corrupted phase map with a processing mask.



(a)



(b)



(c)

Fig. 4. (a) Noisy image of Fig. 3 processed by algorithm A, (b) noisy image of Fig. 3 processed by algorithm B, (c) noisy image of Fig. 3 processed by algorithm C.

the algorithm to unwrap the noisy phase map, shown in Fig. 3. We call them A, B and C. All of them follow the flow diagram of Fig. 1, except in its processing part. In all the examples shown here, the set of neighbors of the integrator pixel are the 8-connected neighbors; that is, $N = 8$. The number of rejected pixels is five; that is, $M = 5$. When the integrator and the current pixel are not neighbors, the set of neighbors used in 24; that is, $L = 24$.

Algorithm A follows entirely the flow diagram of Fig. 1; that is, sorting by local gradient and exclusion of the five higher gradients modulo 2π. The result of unwrapping the corrupted phase map of Fig. 3 is shown in Fig. 4(a).

Algorithm B is the same, but the four highest gradients are not excluded from the process. The sorting by local gradient modulo 2π still remains. The result of unwrapping the corrupted phase map of Fig. 3 is shown in Fig. 4(b).

In algorithm C there is no exclusion of any point from the process and there is no sorting by local gradient modulo 2π. The result of unwrapping the corrupted phase map of Fig. 3 is shown in Fig. 4(c).

As Fig. 4(a) shows, the best performance is achieved by algorithm A. While the phase map is well sampled, the noise is not taken as phase jumps because for a well-sampled phase map the gradient modulo 2π across a phase jump is low. As the sampling frequency tends to the Nyquist frequency, the gradient modulo 2π across a phase jump increases, making the operation of discerning between noise and phase jumps more difficult. If the noisy points are not identified, they are processed and the errors associated with them are propagated across the image, as shown in Figs. 4(b) and 4(c).

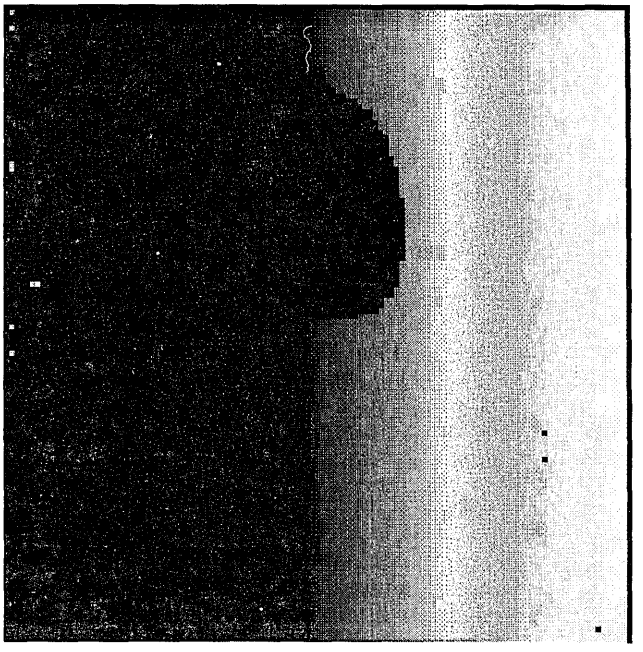
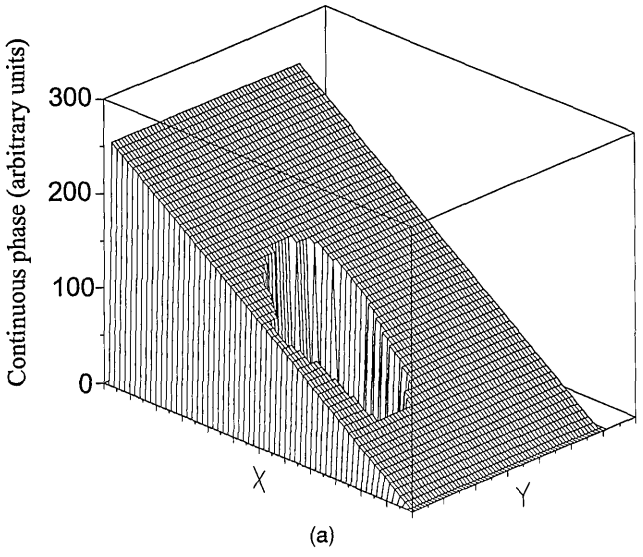
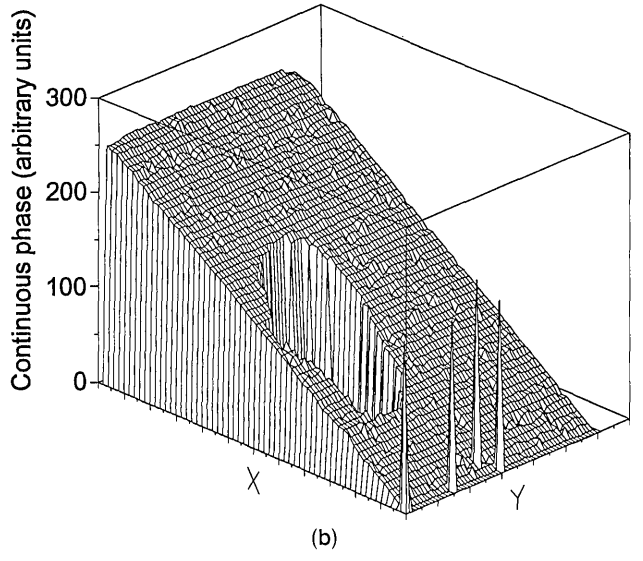


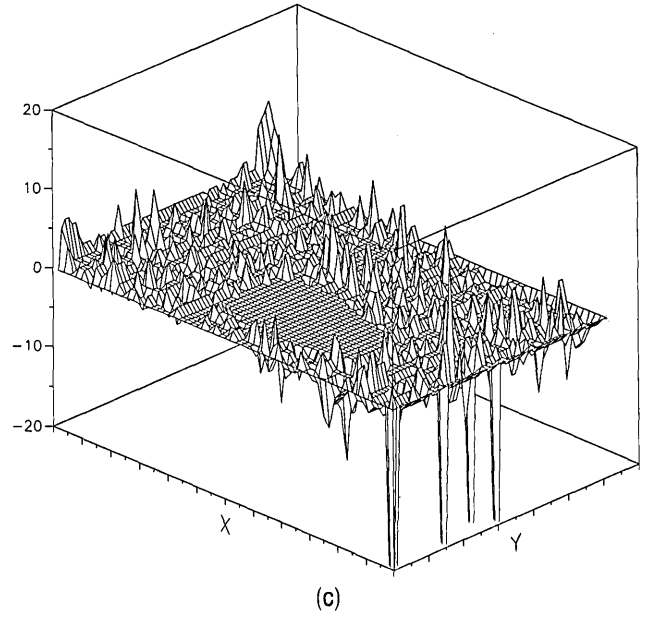
Fig. 5. Result of the bilinear interpolation of the holes of Fig. 4(a).



(a)



(b)



(c)

Fig. 6. (a) Three-dimensional representation of the continuous phase that results from unwrapping the clean phase map of Fig. 3, (b) three-dimensional representation of the continuous phase of Fig. 5, (c) difference between (a) and (b).

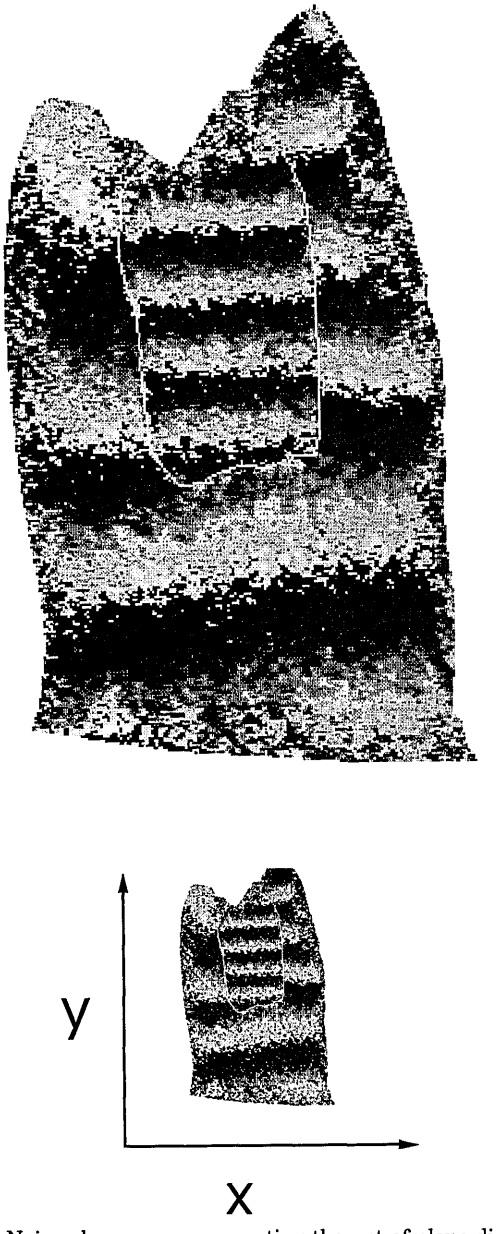


Fig. 7. Noisy phase map representing the out-of-plane displacement of a tooth and its filling measured with a speckle interferometer. The orientations of the X and Y axes are also shown. This orientation is the same for Fig. 8.

The final step is the estimation of the true phase from the image shown in Fig. 4(a). The desired estimation must give a value to the nonprocessed pixels that appear as holes, as in Fig. 4(a). The strategy that we follow is the bilinear interpolation of the nonprocessed points. The result of interpolating Fig. 4(a) is shown in Fig. 5.

Next, the results of processing the clean and the corrupted phase maps are compared. The continu-

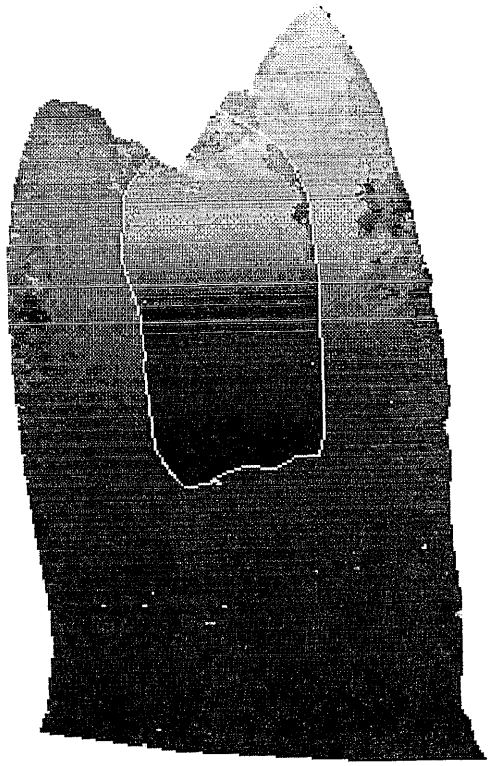


Fig. 8. Complete processing of Fig. 7, that is, unwrapping and further bilinear interpolation of the nonprocessed points. The tooth and the filling are processed separately.

ous phase map that results from unwrapping the clean phase map of Fig. 2 is shown in Fig. 6(a). The three-dimensional representation of Fig. 5 is shown in Fig. 6(b). Figure 6(c) shows the difference between Figs. 6(a) and 6(b). In Fig. 6(c) it is possible to see the main features of the algorithm. Out of the areas affected by the noise the processed data are correct; there is no propagation of errors. In the pixels affected by the noise the errors are limited.

Finally, a noisy speckle interferogram is used to test the algorithm. The object is a tooth with a filling. The phase map representing the out-of-plane displacement is shown in Fig. 7. The phase maps corresponding to the tooth and to the filling are processed separately. In this case the set of neighbors used for the integrator pixel are the 24-connected neighbors, that is, $N = 24$. The number of rejected pixels is 20; that is, $M = 20$. When the integrator and the current pixel are not neighbors, the set of neighbors used is 24; that is, $L = 24$. The result of the complete processing of Fig. 7, that is, unwrapping and further bilinear interpolation of the nonprocessed points, is shown in Fig. 8. The three-dimensional representation of the data of Fig. 8 is shown in Fig. 9. As can be seen, the result is very good for both parts, tooth and filling.

The authors thank H. Steinbichler and all the nice people at Steinbichler Optotechnik GmbH for their support and friendship and also for the image of Fig. 7. This work was partially supported by BRITE/EURAM project 3599, BREU 0120-C, and MAT 91-1389-CE.

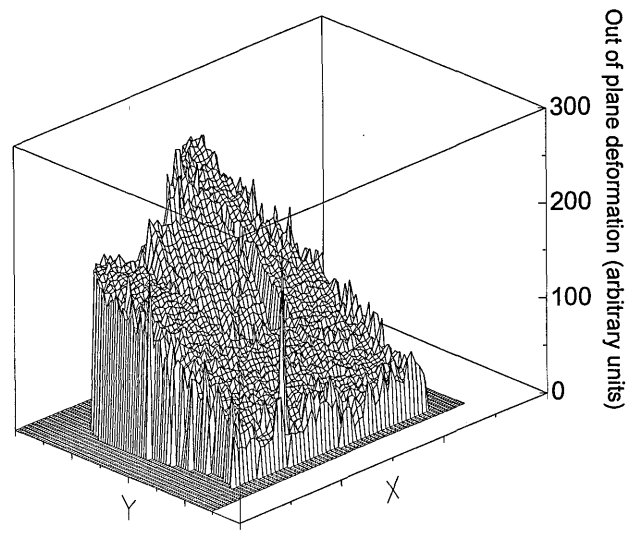


Fig. 9. Three-dimensional representation of the continuous phase map of Fig. 8.

References

1. K. Creath, "Phase-measurement interferometry techniques," Prog. Opt. **26,** 349–393 (1988).
2. J. E. Greivenkamp, "Sub-Nyquist interferometry," Appl. Opt. **26,** 5245–5258 (1987).
3. P. Andrä, U. Mieth, and W. Osten, "Strategies for unwrapping noisy interferograms in phase sampling interferometry," in *Industrial Applications of Holographic and Speckle Measuring Techniques,* W. P. Jueptner, ed., Proc. Soc. Photo-Opt. Instrum. Eng. 1508, 50–72 (1991).
4. D. P. Towers, T. R. Judge, and P. J. Bryaston-Cross, "A quasi heterodyne technique and automatic algorithms for phase unwrapping," in *Fringe Pattern Analysis,* G. T. Reid, ed., Proc. Soc. Photo-Opt. Instrum. Eng. **1163,** 95–119 (1989).
5. J. M. Huntley, "Noise-immune phase unwrapping algorithm," Appl. Opt. **28,** 3268–3270 (1989).
6. D. J. Bone, "Fourier fringe analysis, the two-dimensional phase unwrapping problem," Appl. Opt. **30,** 3627–3632 (1991).
7. D. Ghiglia, G. A. Mastin, and L. A. Romero, "Cellular-automata method for phase unwrapping," J. Opt. Soc. Am. A **4,** 267–279 (1987).
8. J. M. Huntley, R. Cusack, H. Saldner, "New phase unwrapping algorithms," in *Proceedings FRINGE 93,* W. Jüptner and W. Osten, eds. (Akademie, Berlin, 1993), pp. 148–153.
9. K. Itoh, "Analysis of the phase-unwrapping algorithm," Appl. Opt. **21,** 2470–2476 (1982).
10. H. A. Vrooman and A. M. Maas, "Image processing algorithms for the analysis of phase-shifted speckle interference patterns," Appl. Opt. **30,** 1636–1641 (1991).
11. D. Dirksen, X. Su, D. Vukicevic and G. Von Bally, "Optimized phase shifting and use of fringe modulation function for high resolution phase evaluation," in *Proceedings FRINGE 93,* W. Jüptner and W. Osten, eds. (Akademie, Berlin 1993), pp. 72–77.