Phase-unwrapping algorithm based on an adaptive criterion

J. A. Quiroga, A. González-Cano, and E. Bernabeu

A new algorithm for phase unwrapping of phase maps with noise or logical inconsistencies is proposed. It is based on the use of an adaptive threshold and the second difference of the locally unwrapped phase as a selection criterion for the pixels to be processed.

Key words: Phase unwrapping, fringe-pattern processing.

Fig. 1. Neighbor set for a given pixel, 0, and directions for the calculation of the second differences.
way, propagation of errors in the unwrapping process was stopped or at least limited.

The main disadvantages of this algorithm were the following:

1. The fact that the number of points to be rejected is fixed implies that too much valid information can be rejected in noise-free zones of the phase map, whereas too much nonvalid information is considered in noisy zones.

2. The use of local phase gradient modulo $2\pi$ as a selection criterion makes it difficult to detect the zones of high curvature of the phase map that are usually associated with logical inconsistencies or discontinuities resulting in integer fringe shifts.

The algorithm that we present here is designed to deal with these types of problems. As in the previously depicted algorithm, the processing begins with any starting point that becomes the first so-called integrator pixel. We consider the 16 neighbors of this pixel represented in Fig. 1, and we calculate the second difference of the locally unwrapped phase in the directions defined by the eight connected neigh-

![Computer-generated phase map with a discontinuity that produces an integer fringe shift.](image1)
![Phase map of (a) when a linear carrier phase is added.](image2)
![Unwrapped phase map when the second-difference-based algorithm is used. Black points correspond to nonprocessed points.](image3)
![Final unwrapped phase map, when the value of the phase in the nonprocessed points is obtained by interpolation.](image4)
stored. We define a threshold \( T \) and we order the pixels 1–8 according to their second differences. However, not all the neighbors of the first pixel are stored. We order the pixels 1–8 according to their second differences, and we store them in the queue to be processed. Then, the first pixel in the queue (let us denote it by \( p \)) is extracted, and we calculate its unwrapped phase, \( \Phi \), with respect to the integrator pixel, 0, by

\[
\Phi(p) = \Phi(0) + \left| \Phi(p) - \Phi(0) \right|.
\]

Now, we consider \( p \) as a new integrator pixel and proceed again in the way depicted above, ordering its eight connected neighbors according to their second differences. However, not all the neighbors of \( p \) are stored. We define a threshold \( T \) based on the values of the second differences of the neighbors of the first integrator pixel, 0. A neighbor of \( p \) is rejected and subsequently not stored in the queue if the value of the second difference for the direction defined by \( p \) and itself is strictly bigger than \( T \). We may use as \( T \), for instance, the third value in the list of second differences for pixel 0 (this is a good estimation for the median of the set of the eight second differences for 0); any other value, i.e., the second, the fourth, etc., could be used depending on the image. We found that the third is a good compromise between noise management and the size of the processed area of the phase map.

According to this criterion, a certain number of neighbors of \( p \) are rejected, and the remaining ones are stored in the queue. Then, we take the first pixel in the queue, and we calculate its unwrapped phase, as in Eq. 4. We take it as the new integrator pixel, and we recalculate the threshold, \( T \), for the new set of neighbors. We repeat these operations obviously avoiding processing of previously processed pixels until there are no pixels in the queue.

As can be seen, instead of using a fixed number of rejected pixels, we use a threshold that may vary in each step. The variation of the threshold becomes imposed by the local characteristics of the phase map because the value of \( T \) is obtained from the behavior of the phase in a zone near the pixel in which the unwrapped phase is calculated. This means that the algorithm adapts itself to the local characteristics of the area that is processed and thus it overcomes the problem of considering too many or too few pixels in zones with different amounts of noise. Also, the use of the second difference as the selection criterion instead of the local gradient modulo \( 2\pi \) lets one detect and avoid the areas of high curvature.

To illustrate the way that the algorithm works, we generated a simulated phase map that contains a discontinuity in the phase field (Fig. 2a). Figure 2b shows the phase map when a linear carrier phase is added. As can be seen, the discontinuity reaches values bigger than \( 2\pi \) rad one fringe, thus producing an integer fringe shift. The discontinuity can be detected with our algorithm even at the place in which an integer fringe shift appears because in this place there still exists a small change in the curvature of the phase. In this way we can produce an error-free continuous phase map, shown in Fig. 2c. The black points of Fig. 2c correspond to non-processed points whose phase values can be interpolated from the values of their neighbors. The result of this process is shown in Fig. 3.

![Image](image_url)
interpolation (i.e., the final unwrapped phase map is shown in Fig. 2d).

In a phase map with problems of the type discussed, an algorithm based on first differences is not effective because it cannot detect the discontinuity when it produces an integer fringe shift. Also, as Bone shows, a branch-cut method is not advisable because the number of positive and negative discontinuity sources is not the same in the conflictive region, which results in isolated discontinuity sources far from the boundary. This means that the isolated discontinuity sources must be joined to the border of the map, thus producing an incorrect distribution of cut lines. The computing time of our method was approximately 3 min. on a PC 486 66-MHz microcomputer for a size of 128 × 128 pixels. A branch-cut method followed by a simple unwrapping algorithm can be faster, but as we have said, it cannot be applied in this case. The election of our algorithm or a branch-cut method will then depend on the phase map being processed.

Figure 3 shows the results of the method in a real case. The phase map [Fig. 3a] presents two types of problems: a zone of high curvature (marked with 1) and a 2π jump hidden by noise (zone 2). As can be seen in the unwrapped phase map shown in Fig. 3b, the algorithm can overcome both problems. The use of an adaptive threshold is efficient in cases such as this one, in which regions of different characteristics are to be processed. The computing time, including interpolation of nonprocessed points, was ~5 min. on a PC 486 66-MHz microcomputer.

We thank Hans Steinbichler for the images of Fig. 3. This work was partially supported by Tecnologias Avanzadas de la Produccion under project TAP92-0087 from the Comision Interministerial de Ciencia y Tecnologia.

References


