applied optics

Fast quality-guided phase unwrapping algorithm through a pruning strategy: applications in dynamic interferometry

Lourdes López-García,[1] Anmi García-Arellano,[2] and William Cruz-Santos[1,*]

[1]CU-UAEM Valle de Chalco, Hermenegildo Galeana 3, Valle de Chalco 56615, Estado de México, Mexico
[2]CONACYT-ECOSUR, Unidad Chetumal, Av. del Centenario Km. 5.5, Chetumal, Quintana Roo 77014, Mexico
*Corresponding author: wdelacruzd@uaemex.mx

Received 30 January 2018; revised 7 March 2018; accepted 19 March 2018; posted 19 March 2018 (Doc. ID 321052); published 17 April 2018

The quality-guided phase unwrapping algorithm is one of the most employed spatial algorithms due to its computational efficiency and robustness. It uses a quality map to guide the unwrapping process such that pixels are processed according to their quality values from highest to lowest. Several improvements have been proposed during the last few years with the purpose of using it in time-demanding applications. However, many of the proposals depend on the distribution of the values on the given quality map. In this paper, a novel pruning strategy based on a red–black tree data structure is proposed, whose complexity time is independent of the distribution of the given quality map. We take advantage of the partial ordering of the branches in a red–black tree together with a pruning strategy to speed up the unwrapping process. Experimental results, using real and simulated data, show that the complexity time of our proposal improves the existing quality-guide-based algorithms. Also, a series of interferometric patterns of a time-varying phase distribution experiment have been processed showing that our proposal can be used for real-time applications. The source code of the implemented algorithms is publicly available.

OCIS codes: (120.0120) Instrumentation, measurement, and metrology; (100.5070) Phase retrieval; (100.5088) Phase unwrapping; (120.3180) Interferometry.

https://doi.org/10.1364/AO.57.003126

1. INTRODUCTION

In phase measuring techniques, such as phase-shifting interferometry, phase-shifting profilometry, and synthetic aperture radar (SAR) [1,2], a *wrapped phase* is retrieved, which is characterized by its 2π discontinuities. The *phase unwrapping* problem consists in removing these discontinuities from the wrapped phase to obtain a continuous phase map. There exist several unwrapping algorithms that can be grouped as *spatial* and *temporal*. Spatial algorithms are based on a path integration strategy to obtain the unwrapped phase, and the temporal algorithms use a sequence of temporal wrapped phase maps to recover the absolute unwrapped phase map. Among the spatial algorithms are the Goldstein algorithm [3], the quality-guided path following algorithm [4], the mask cut algorithm, and the Flynn's algorithm [5]. On the other hand, in the temporal algorithms are the gray-code temporal phase unwrapping algorithm [6], the multifrequency algorithm [7], and the multiwavelength or heterodyne approach [8].

Over the last years, the incorporation of high-speed cameras [9,10], fast-pulsed lasers [11], and fast digital video projectors [12] in phase measuring techniques (among other electronic instrumentation) has afforded the ability to capture several hundreds of phase-shifted patterns in short periods of time, principally for different phase measurements of dynamic applications [13–18] for the most time-consuming tasks in order to retrieve the desired phase concerns to the phase unwrapping process. Thus, in order to go hand in hand with the increases in speed of the phase measurement systems, more efficient phase unwrapping algorithms are required.

The quality-guided phase unwrapping (QGPU) algorithm is one of the most used due to its ease of implementation. It is based on a quality map that serves to quantify the reliability of each pixel in the wrapped phase and is used to guide the unwrapping process such that those pixels with high quality are processed first, and those pixels with low quality are processed later. There are several quality map definitions such as the pseudo-correlation coefficient map that appears in SAR and phase derivatives maps [19]; besides, there are methods based on the Fourier transform [20], the windowed Fourier filtering [21,22], and the wavelet transform method [23–25]. In the QGPU algorithm, a data structure called the *adjoin list*, is used, which consists of a sorted list that serves to enqueue pixels with respect to their quality values. The adjoin list supports two operations, the *insert* operation, which adds an element to the list with respect to its quality value, and the *delete* operation, which removes the element with highest quality in the list. Generally, the insert operation is time-consuming, and the delete operation can be done in constant time.

Several authors have proposed improvements to the QGPU algorithm. Ghiglia and Pritt [19] proposed to use a 1D array to implement the adjoin list, where the insert operation can be done in linear time with respect to the length of the array, and the delete operation can be done in constant time by removing the first element in the array. Further improvements [26] propose to use a linked-list (LL) data structure whose complexity of the insert and delete operations have the same complexity as an array. Other authors [22,27] suggest using a partition of the range of the quality values, called I2L2, where, for each interval in the partition, there is an independent adjoin list. The idea behind this approach is based on the observation that the complexity of the insert operation is more efficient when the list contains few pixels. The main disadvantage of this partition strategy is that some intervals could contain many pixels if the distribution of the quality values is not well balanced, which is the case of most quality maps.

In this paper, we propose a new method to implement the adjoin list in the QGPU algorithm, which improves the data structures based on LL and I2L2. Our method takes advantage of the partial ordering of the branches of balanced search trees together with a pruning strategy of the tree to process the pixels according to their quality values. The proposed pruning strategy allows us to reduce the complexity of the insert operation by maintaining the height of the subtrees as small as possible. Experimental evaluation of our proposal using three kinds of quality maps show that the complexity of the pruning strategy is independent of the distribution of the quality values. Also, we consider the efficiency of our QGPU algorithm implementation in a dynamic scenario, which requires the computation of the unwrapped phase for a large number of wrapped phase maps obtained from a high-speed camera used in a previously investigation [28]. Our obtained results in this dynamic scenario show that the pruning strategy improves considerably the computation time with respect to the LL and I2L2 methods.

The organization of this paper is as follows: in Section 2, the proposed pruning strategy is given; in Section 3, a comparison and experimental evaluation is shown; and Section 4 discusses conclusions and future research.

2. PRUNING STRATEGY

The QGPU algorithm uses a matrix that contains the quality values for each pixel in the wrapped phase map. The algorithm begins processing the pixels with the highest value of quality and continues unwrapping its neighbor pixels according to the quality values, until covering all the wrapped phase. Each processed quality value is stored in order to obtain the path integration. At the end, the algorithm reports the unwrapping phase map and its path integration.

In this context, an optimum process is to sort the quality map to guarantee that the path integration is created from the highest value to the lowest value of quality; however, this is inefficient. Thus, some strategies have been proposed using programming tools such as linked list and binary trees [19,22,27].

Considering that the wrapped phase map has $N \times M$ pixels to process, a sorted linked list will have an algorithmic complexity in the insertion function of $O(N \times M)$. To reduce execution time, the linked list is bounded with z elements to insert. When a threshold is reached, the linked list is divided into two parts; the half list with the minor values is pasted in other linked list, and the other half of the list is left pending to process. Then, the insertion process is realized with better execution time. Nevertheless, the adjoin list grows $z/2$ elements each time that the threshold is reached; thus, the initial problem reappears.

On the other hand, in binary trees, the insertion is more efficient than the insertion in the linked list. A binary tree is generated with nodes, where each node contains three elements: a data, a left reference, and a right reference. The first element inserted in the tree is called the *root*, and every node in the binary tree is linked with, at the most, two nodes. A node with one or two references is called *father*; in the other case, it is called *leaf*.

The insertion operation adds a new node in an empty reference called NULL. On the left side, if the value of the new node is less than or equal to its father or in the right side, otherwise. It is important to note that, if there is a small root, the tree can be unbalanced to the right side; if there is the big root, then the tree can be unbalanced to the left side.

Fortunately, the red–black trees (RBT) [29] are special kinds of binary trees that allow the insertion or deletion functions to balance in each operation in a tree through nodes classified in red or black. Figure 1 shows the difference between an unbalanced binary tree and its red–black tree version. The algorithmic complexity of the insertion and deletion functions is $O(\log n)$, where n is the number of nodes in the tree, and the execution time is reduced due to the tree being balanced.

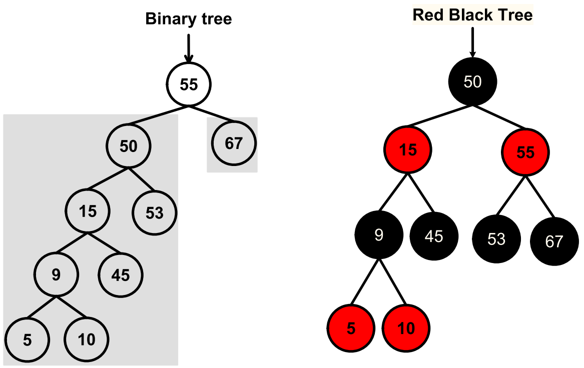

Fig. 1. Difference between an unbalanced binary tree and its red–black tree version with the respective left and right rotations.

Taking advantage of a balanced tree, it is possible to divide the RBT in two subtrees of a similar number of nodes. Each level of the tree, denoted by l, has 2^l number of nodes, and the RBT allows the insertion function to produce a tree with just one level of difference between of the both subtrees. In the best case, an RBT with n internal nodes (without the root) has $n/2$ nodes in each subtree, whereas in the worst case, there are $2^l - (2^{l-1})$ nodes of difference.

As can be seen in Fig. 2, the RBT has five nodes on the left side and three nodes on the other side. Then, the steps to divide the tree are as follows: (1) a pointer called *left_tree* is placed over node 15; (2) a pointer called *right_tree* is placed over node 55; (3) the node 50 is separated from the tree, obtaining three trees; (4) node 50 is inserted in the *left_tree*; and (5) two trees are obtained.

The division of the tree is called *pruning*. An important fact is that the insertion is sorted, which means the highest value is always on the right side of the tree. Specifically, the last node on the far-right branch is the biggest number in the tree. With this benefit, the unwrapping map can be calculated following the quality values inserted in an RBT. To avoid the tree growing the same length of the phase map, in this paper, pruning of the RBT is proposed. The process consists of counting the number of inserted nodes; when it is z, the pruning is done, and the left subtree of an approximate length of $z/2$ is inserted in a list of trees. The pruning makes a difference between the adjoin list and the list of trees, due to each tree inserted is already ordered, whereas the insertion in the adjoin list must be sorted. Because the position of the highest node is always the last node on the far-right branch, the searching of this one is not necessary.

It is important to note that the list of trees contains k number of RBTs, where each tree has $z/2 + 1$ nodes, approximately. If the length of the wrapped phase map is $N \times M$, the first pruning takes z of those elements beginning with the highest value of quality. The next pruning requires $z/2$ elements to complete the threshold z. Thus, there are $(N \times M - z)/z + 1$ pruning, calculated. If z is big, it implies that each tree in the list grows in depth with $2^l \approx N \times M$ when $k = 1$; then, the time execution of the insertion and deletion functions would be expensive. If z is small, then k would be large, and the list of trees will contain small trees. At times, each small tree in the list is processed; taken its highest value as a new start point, this fact causes many jumps among the quality values. Thus, the unwrapping phase map loses quality by not considering some areas of the wrapped phase.

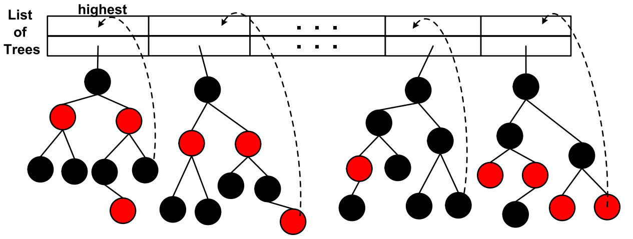

Fig. 3. List of RBT trees, generated by the pruning of the main RBT.

We propose the value of z experimentally as $z = 2 \cdot \min\{N, M\}$, where N and M are the height and width of the given phase map, respectively. With this value z, we have that the number of nodes in each tree in the list is at most $z/2 + 1$; consequently, the length of the list is at most $\max\{N, M\} - 1$. Besides, the depth of the tree decreases to $2^{l-1} \approx \min\{N, M\}$ getting to decrease the time execution of the insertion, deletion, and balanced functions in the tree.

Figure 3 shows the list of RBTs, each one with the direction of the highest-quality value. In the phase map, when a pixel does not have a neighbor without processing, the first tree in the list is taken. It is important to note that this tree has the highest values found in the first pruning. The process is repeated until the tree list is empty. A significative improvement is that the trees in the list are sorted, and the highest values of each tree are identified. Thus, the execution time in the insertion and the deletion of the highest value is widely reduced, allowing the unwrapping phase algorithm to be more efficient.

The unwrapping phase process is shown in Algorithm 1. The pruning of the RBT is realized when the tree reaches z number of nodes. The left side contains all small values less than the root, and it is included in the list of trees, whereas the right side is taken as the main RBT. Once the RBT is empty, the first tree in the list is taken. As it can be observed, as the phase map is covered, the number of insertions is reduced, until the list is empty. The pruning is presented in Algorithm 2.

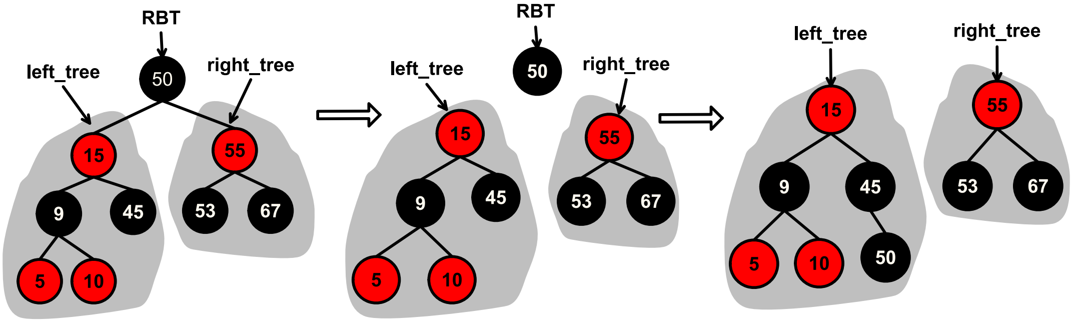

Fig. 2. Example of a pruning of the RBT.

Algorithm 1: Quality-guide Phase Unwrapping Algorithm

```
procedure QGPU
    Find a starting pixel with high-quality value
    Store its phase value in the solution array
    Mark the pixel as unwrapped
    Insert the pixel in the RBT
    while RBT is not empty do
        Remove the highest-quality pixel from the RBT
        for each of their neighbor pixels do
            if the pixel is not unwrapped then
                Unwrap the pixel and store its value
                Check RBT insertion
                Mark the pixel as unwrapped
                if RBT is empty then
                    Start pruning
```

Algorithm 2: Pruning Algorithm

```
procedure pruning
    if quality value is highest than high value then
        Update high value
    if number of nodes in RBT is highest than threshold then
        Point to the left child
        Point to the right child
        Assign the right child to RBT
        Store the left child in the tree list
        Insert the root to the left tree as the highest node
        Reset threshold of RBT
```

3. EXPERIMENTS AND DISCUSSION

This section presents an experimental evaluation of the proposed pruning strategy in comparison with the LL and I2L2 approaches. We first explain the system configuration and data preparation to test the performance of the implementations of the QGPU algorithm. Later, we evaluate and discuss the proposed pruning strategy in different static and dynamic scenarios.

A. Data Preparation and System Specification

In order to test some static phase measurements, we considered three wrapped phase and quality maps: the first one is a simulated wrapped phase map of size 512 × 512 pixels using the built-in function peaks in MATLAB with white Gaussian noise and a quality map based on maximum gradient, as shown in Figs. 4(a) and 4(b). The second one is a wrapped phase map of size 307 × 567 pixels of a small rhino object obtained with recent one-shot interferometric techniques employing coherent illumination and the phase-shifting method of modulation of polarization [28] and filtered amplitude [30] as a quality map, as shown in Figs. 4(d) and 4(e). The last one is a wrapped phase map of size 1404 × 1220 pixels obtained from the digital projection of fringe patterns on a plate and modulation data as the quality map shown in Figs. 4(g) and 4(h). The above quality maps already described were represented using single floating-point precision.

All the experiments were performed on a desktop computer using the Linux Ubuntu operating system with an Intel Core i7-6700T processor at 2.8 GHz and 8 GB RAM. The programming language used was ansi C using the gcc compiler. The computed execution time in the implementations of the QGPU algorithm does not include the calculation of the wrapped phase and quality maps. The comparisons shown in this section of the pruning strategy with respect to the LL and

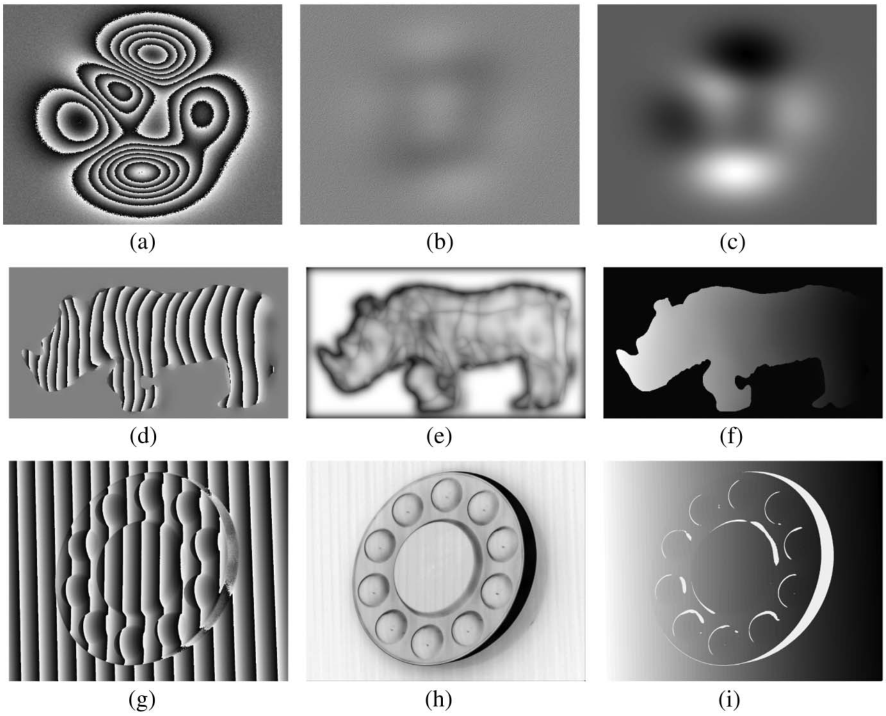
(a) (b) (c)
(d) (e) (f)
(g) (h) (i)

Fig. 4. Wrapped phase, quality map, and unwrapped phase. (a)–(c) Peaks phase map, maximum gradient quality map, and unwrapped phase. (d)–(f) Filtered phase, filtered amplitude, and unwrapped phase. (g)–(i) Phase-shifting map, modulation quality map, and unwrapped phase. All the data described above have been normalized and scaled for visualization purposes.

I2L2 approaches were achieved using our own optimized implementations. The implementation of algorithm I2L2 was based on reference [27]. The source code of the implemented QGPU algorithms can be downloaded at (https://goo.gl/udBRcE).

B. Static Evaluation

A comparison of the execution time for the pruning strategy and the LL and I2L2 approaches is shown in Table 1. The list size of the adjoin list in the LL approach was set to $N + M$, as suggested by Ghiglia and Pritt [19], where N and M are the height and width of the phase map, respectively. The number of partitions or *level number* used in the I2L2 was set to $(NM)^{1/2}$, as in Zhao and Kemao [27]. For the pruning strategy, the threshold value z was set to $2\ \min\{N, M\}$, as explained in Section 2. As shown in Table 1, the minimum execution time for the phase data in Fig. 4 is for the pruning strategy, while the LL and I2L2 approaches obtained higher execution times for the three phase maps. The speedup of the pruning strategy with respect to the LL (I2L2) approach for the peaks, rhino, and plate phase maps was 19 (5.8), 2.89 (3.79), and 20.32 (14.67) times faster, respectively. The obtained unwrapped phase maps by the pruning strategy for the peaks, rhino, and plate wrapped phase are shown in Fig. 4(c), 4(f), and 4(i), respectively.

Figure 5 shows the complexity of the insert operation in the pruning strategy and the LL and I2L2 approaches. As shown in Fig. 5, the execution time of the pruning strategy is the lowest one and is almost the same for the peaks, rhino, and plate wrapped phase maps. However, the execution time for LL and I2L2 is greater than the pruning strategy, and they also vary with respect to the given quality map. To understand this behavior, one can notice that the complexity of the insert operation depends on the quality value assigned to each pixel in order to find a proper position into the adjoin list. Figures 6(a) and 6(b) show the histogram and cumulative distribution function (CDF), respectively, for the peaks, rhino, and plate quality maps.

From the CDF shown in Fig. 6(b), one can notice two important intervals of the quality values, the first one between 0 and 0.49 and the second one between 0.49 and 1.0. In the first interval, the probability of inserting a pixel with a quality value between 0 and 0.49 is approximately 0.05 for the three quality maps, which also means that the probability of inserting a pixel with a quality value greater than 0.49 is 0.95. In contrast, the CDF for the rhino quality map is almost linear between 0.49 and 1.0 in which the lowest execution time of the insert operation is obtained, as shown in Fig. 5. Finally, in the peaks and plate quality maps, the CDF is not linear between 0.49 and 1.0 in which we observe an increase in the execution time of the insert operation.

The previous analysis shows us that the complexity of the insert operation depends on the quality data distribution for the LL and I2L2 phase unwrapping algorithms. On the other hand, the complexity of the pruning strategy is independent of the distribution of the quality maps. The latter can be explained because the theoretical complexity of the insert in a RBT is $O(\log n)$, where n is upper bounded by $2 \times \min\{N, M\}$; in practice, this complexity is almost constant.

We also perform a comparative error analysis to validate the accuracy of the pruning strategy. We calculated the mean

Table 1. Execution Time of the LL, I2L2, and Pruning Algorithms Using the Data Shown in Fig. 4[a]

		Time (ms)		
Name	Resolution	LL	I2L2	Pruning
Peaks	512 × 512	945.81	288.0	49.56
Rhino	307 × 567	101.02	132.2	34.87
Plate	1404 × 1220	6916.54	4995.25	340.34

[a] Execution times are given in milliseconds and are averaged over 20 repetitions.

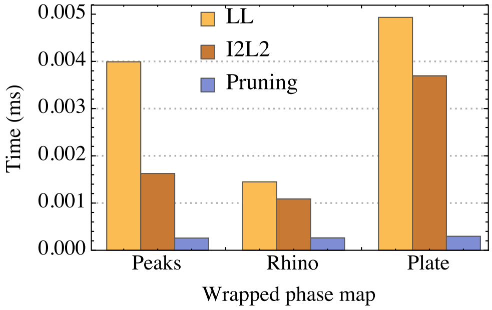

Fig. 5. Comparison of the execution time of the insert operation for the pruning strategy, LL, and I2L2 approaches using the wrapped phase and quality maps given in Fig. 4. The average execution time of all the calls to the insert operation over the complete process of the unwrapping algorithm are shown.

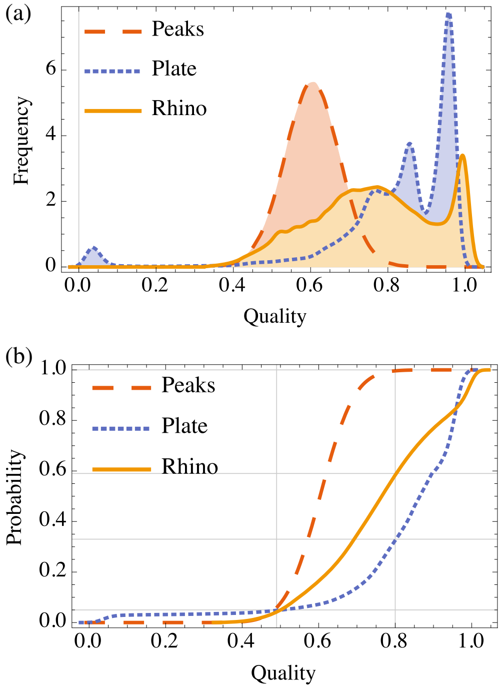

Fig. 6. (a) Histogram and (b) cumulative distribution for the peaks, rhino, and plate quality maps, as shown in Fig. 4. For the three cases, the quality data have been normalized and interpolated.

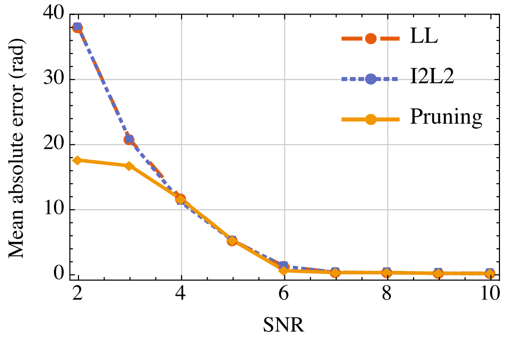

Fig. 7. Error analysis comparison between algorithms LL, I2L2, and pruning.

absolute error (MAE) between simulated peaks distributions of size 512 × 512 pixels and the unwrapped phase maps obtained with LL, I2L2, and pruning. The input of the algorithms consists of wrapped peak distributions with a white noise level and quality map based on the maximum gradient. In order to make the absolute difference between the ideal phase map and the relative phase maps obtained by the unwrapping algorithms, all the phase values of the relative maps were displaced with respect to the ideal phase value of the initial point of the unwrapping process. Finally, the absolute average of all absolute differences was calculated. Figure 7 shows the MAE for several signal-to-noise (SNR) ratios of the white noise added to the simulated peak distributions. As shown in Fig. 7, the MAE is smaller for the pruning algorithm when the level of noise is higher or SNR = 2,3, and it has a similar MAE with respect to LL and I2L2 algorithms, when the noise level decreases or SNR = 3, …, 10. All the experiments were carried out with a value of $z = 2 \times 512$ for the pruning algorithm in which we found the best trade-off between efficiency and accuracy.

Figure 8 shows a numerical analysis for the selection of the z parameter in the pruning algorithm. We considered the simulated peak distribution of size 512 × 512 pixels with added noise level SNR equal to 7 and a quality map based on the maximum gradient. We calculate the MAE and execution time of the pruning algorithm for several values of parameter z. The values for $z = f \cdot \min\{N, M\}$, where $f = 0.25, 0.5, 0.75, 1, 2, 3, 5$ and N, M, are the height and width of the wrapped phase map, respectively. As can be seen in Fig. 8, when the parameter z is lower than 2×512, the execution time decreases and the MAE increases. On the other hand, when z is greater than 2×512, the execution time increases, and the MAE remains almost constant. This numerical study using the simulated peak distribution shows that the selected value of $z = 2 \cdot \min\{N, M\}$ is optimal with respect to the MAE and execution time.

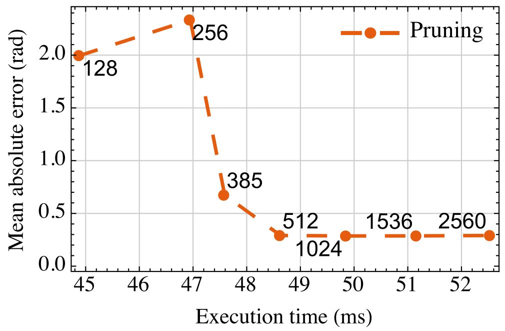

Fig. 8. Numerical analysis for the z parameter selection in the pruning algorithm. For each selected value of z (labels in the dashed line), the execution time and the MAE are shown.

C. Dynamic Evaluation

For our dynamic evaluation, we processed some series of interferometric one-shot phase-shifted patterns, obtained with a previously reported optical system mainly designed for studying applications involving time-varying optical phase distributions based on modulation of polarization with pulsed lasers and fast cameras [28,31]. Figure 9 shows some representative experimental results obtained in one shot, with the mentioned optical system at an operation rate of 500 frames per second (fps). Figure 9(a) is part of a first interferometric optical evolution generated by a constant wavefront tilt and by completely inserting an ethanoate sample of size 3 × 3 cm in one of the arms of the interferometric arrangement of the mentioned optical system [28,31]. This dynamic evolution (Serie 1) was registered in a sequence of 9944 images, each one at 93 × 93 pixels. An animation of one-shot interferograms at different instances of time of this dynamic evolution is presented in Visualization 1; meanwhile, Visualization 2 shows the corresponding dynamic unwrapped phase, where, as can be observed, some discontinuities appear, caused mainly by the application of a slight manual mechanical stress to the ethanoate sample during the interferometric test generating abrupt changes in the optical phase. The execution times of Serie 1 are presented in Table 2. Figure 9(b) shows a second interferometric optical evolution (Serie 2) in a sequence of 9994 images, each one at 97 × 97 pixels, this time generated by a constant optical step obtained by inserting half of the same ethanoate samples in one of the arms of the interferometric arrangement of the optical system. Visualization 3 shows the animation of some interferograms at different instances of time for this new optical evolution and Visualization 4 shows its corresponding dynamic unwrapped phase. The execution times of this evaluation are presented in Table 2. Finally, Fig. 9(c) shows our third evaluation, which consisted of processing a sequence of 11,600 images (Serie 3), each one a size of 79 × 79 pixels, corresponding to a variable optical step generated with the mentioned ethanoate sample. Visualization 5 and Visualization 6 show a sample of the interferograms at different instances of time and their corresponding dynamic unwrapped phase, respectively; the execution times of this test are also presented in Table 2.

The results shown in Table 2 were obtained using the system specifications mentioned in Section 3.A. The captured frames with the fast camera have a resolution of 768 × 224 pixels and the region of interest (ROI) per interferogram is at most 97 × 97 pixels. The wrapped phase was obtained using a four-step phase-retrieval algorithm [28], and it is used as the amplitude of the wrapped phase as a quality map [21]. Both the wrapped phase and quality maps were computed and stored before using our QGPU algorithms. As shown in Table 2, the

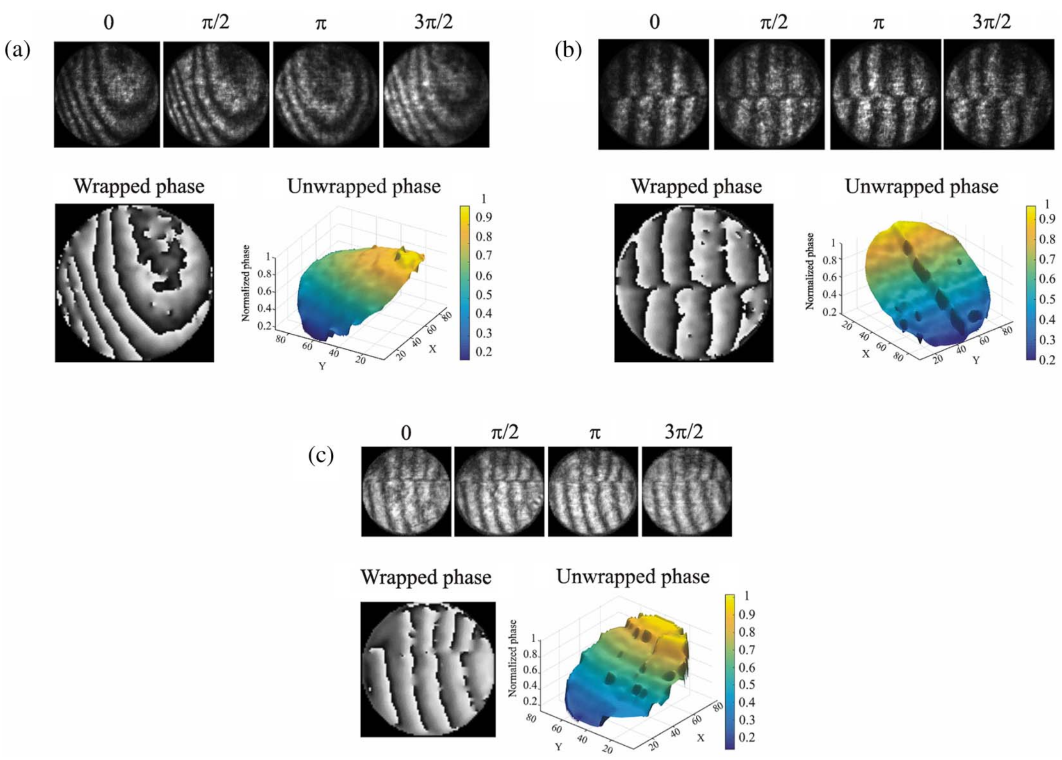

Fig. 9. Representative one-shot phase-shifted patterns employed in our dynamic evaluation generated in a dynamic optical system based on a Mach–Zehnder interferometer. The phase-shifts are shown at the top of each figure. (a) Optical results of a constant wavefront tilt and an ethanoate sample and its corresponding wrapped and unwrapped phase. (b) Optical step generated by an ethanoate sample and its corresponding wrapped and unwrapped phase are shown. (c) Variable optical step generated by an ethanoate sample is shown, along with its corresponding wrapped and unwrapped phase.

Table 2. Comparison of the LL, I2L2, and Pruning Algorithms in Order to Process a Series of Phase Maps[a]

			Time (s)		
Name	Resolution	Sequence	LL	I2L2	Pruning
Serie 1	93 × 93	9,944	36.3	30.93	12.51
Serie 2	97 × 97	9,944	37.95	33.18	13.17
Serie 3	79 × 79	11,600	24.92	22.82	10.12

[a] Execution times are given in seconds and are averaged over 20 repetitions.

pruning QGPU algorithm is up to three times faster than the LL and I2L2 approaches, and it can process up to 1146 interferograms per second with a resolution of 79 × 79 pixels. It can be noted that, if the ROI grows, then the speed up of the pruning algorithm improves significantly, as shown in the static evaluation in Section 3.B.

4. CONCLUSIONS

We have proposed a new QGPU algorithm using RBTs. The algorithm is based on a pruning strategy of the RBT, with which it generates a list of subtrees of $z/2$ nodes, approximately. Each tree in the list will be processed according to the quality map. Because, the pruning guarantees that each tree in the list is balanced, ordered, and with a similar number of nodes, the distribution of the quality map does not saturate any of the trees. Thus, this proposal improves the complexity of the insert operation in the adjoin list with respect to the LL and I2L2 approaches.

We have evaluated the pruning algorithm for static phase maps, which shows that the distribution of the quality values and the resolution of the wrapped phase maps affects the time execution of the LL and I2L2 approaches, whereas our proposal keeps short execution times, due to it being independent of the distribution of the given quality maps. We have also demonstrated experimentally that the execution time of the insert operation in the pruning algorithm improves the LL and I2L2 methods. In the dynamic evaluation, we processed some series of interferometric one-shot phase-shifted patterns with a resolution at most of 97 × 97 pixels and sequences up to 11,600 images. The execution times presented show that our proposal can be used in real-time applications.

Further research will be focused on the use of multicore architectures and parallel programming in order to speed up the processing of the list of trees generated by the pruning strategy. Such improvements allow us to apply spatial phase unwrapping algorithms such as the QGPU in high-speed 3D imaging systems.

Acknowledgment. A. García-Arellano expresses his gratitude to the National System of Research (SNI - CONACYT) for partial support through grant # 58412.

REFERENCES

1. Z. Malacara and M. Servn, *Interferogram Analysis for Optical Testing*, 2nd ed., Optical Engineering (Taylor & Francis, 2010).
2. D. Malacara, *Optical Shop Testing*, Wiley Series in Pure and Applied Optics (Wiley, 2007).
3. R. M. Goldstein, H. A. Zebker, and C. L. Werner, "Satellite radar interferometry: two-dimensional phase unwrapping," Radio Sci. **23**, 713–720 (1988).
4. H. Lim, W. Xu, and X. Huang, "Two new practical methods for phase unwrapping," in *Geoscience and Remote Sensing Symposium IGARSS 'Quantitative Remote Sensing for Science and Applications', International* (IEEE, 1995), pp. 196–198.
5. T. J. Flynn, "Two-dimensional phase unwrapping with minimum weighted discontinuity," J. Opt. Soc. Am. A **14**, 2692–2701 (1997).
6. G. Sansoni, S. Corini, S. Lazzari, R. Rodella, and F. Docchio, "Three-dimensional imaging based on gray-code light projection: characterization of the measuring algorithm and development of a measuring system for industrial applications," Appl. Opt. **36**, 4463–4472 (1997).
7. J. M. Huntley and H. Saldner, "Temporal phase-unwrapping algorithm for automated interferogram analysis," Appl. Opt. **32**, 3047–3052 (1993).
8. Y.-Y. Cheng and J. C. Wyant, "Two-wavelength phase shifting interferometry," Appl. Opt. **23**, 4539–4543 (1984).
9. A. J. Moore, D. P. Hand, J. S. Barton, and J. D. C. Jones, "Transient deformation measurement with electronic speckle pattern interferometry and a high-speed camera," Appl. Opt. **38**, 1159–1162 (1999).
10. J. M. Kilpatrick, A. J. Moore, J. S. Barton, J. D. C. Jones, M. Reeves, and C. Buckberry, "Measurement of complex surface deformation by high-speed dynamic phase-stepped digital speckle pattern interferometry," Opt. Lett. **25**, 1068–1070 (2000).
11. T. Cookson, J. Butters, and H. Pollard, "Pulsed lasers in electronic speckle pattern interferometry," Opt. Laser Technol. **10**, 119–124 (1978).
12. S. S. Gorthi and P. Rastogi, "Fringe projection techniques: whither we are?" Opt. Lasers Eng. **48**, 133–140 (2010).
13. P. S. H. Song Zhang, "High-resolution, real-time three-dimensional shape measurement," Opt. Eng. **45**, 123601 (2006).
14. S. Zhang, D. V. D. Weide, and J. Oliver, "Superfast phase-shifting method for 3-D shape measurement," Opt. Express **18**, 9684–9689 (2010).
15. N. L. Karpinsky, M. Hoke, V. Chen, and S. Zhang, "High-resolution, real-time three-dimensional shape measurement on graphics processing unit," Opt. Eng. **53**, 024105 (2014).
16. H. Nguyen, D. Nguyen, Z. Wang, H. Kieu, and M. Le, "Real-time, high-accuracy 3D imaging and shape measurement," Appl. Opt. **54**, A9–A17 (2015).
17. D. Blinder, H. Ottevaere, A. Munteanu, and P. Schelkens, "Efficient multiscale phase unwrapping methodology with modulo wavelet transform," Opt. Express **24**, 23094–23108 (2016).
18. O. Backoach, S. Kariv, P. Girshovitz, and N. T. Shaked, "Fast phase processing in off-axis holography by cuda including parallel phase unwrapping," Opt. Express **24**, 3177–3188 (2016).
19. D. Ghiglia and M. Pritt, *Two-Dimensional Phase Unwrapping: Theory, Algorithms, and Software* (Wiley, 1998).
20. X. Su and W. Chen, "Fourier transform profilometry: a review," Opt. Lasers Eng. **35**, 263–284 (2001).
21. Q. Kemao, W. Gao, and H. Wang, "Windowed Fourier-filtered and quality-guided phase-unwrapping algorithm," Appl. Opt. **47**, 5420–5428 (2008).
22. M. Zhao, L. Huang, Q. Zhang, X. Su, A. Asundi, and Q. Kemao, "Quality-guided phase unwrapping technique: comparison of quality maps and guiding strategies," Appl. Opt. **50**, 6214–6224 (2011).
23. Q. Kemao, "Windowed Fourier transform for fringe pattern analysis," Appl. Opt. **43**, 2695–2702 (2004).
24. S. Li, W. Chen, and X. Su, "Reliability-guided phase unwrapping in wavelet-transform profilometry," Appl. Opt. **47**, 3369–3377 (2008).
25. S. Li, X. Wang, X. Su, and F. Tang, "Two-dimensional wavelet transform for reliability-guided phase unwrapping in optical fringe pattern analysis," Appl. Opt. **51**, 2026–2034 (2012).
26. H. Zhong, J. Tang, S. Zhang, and M. Chen, "An improved quality-guided phase-unwrapping algorithm based on priority queue," IEEE Geosci. Remote Sens. Lett. **8**, 364–368 (2011).
27. M. Zhao and Q. Kemao, "Quality-guided phase unwrapping implementation: an improved indexed interwoven linked list," Appl. Opt. **53**, 3492–3500 (2014).
28. A. García-Arellano, W. Cruz-Santos, G. García-Arellano, and J. Rueda-Paz, "One-shot shape measurement of small objects with a pulsed laser and modulation of polarization," Opt. Eng. **56**, 064102 (2017).
29. T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson, *Introduction to Algorithms*, 2nd ed. (McGraw-Hill, 2001).
30. Q. Kemao, W. Gao, and H. Wang, "Windowed Fourier filtered and quality guided phase unwrapping algorithm: on locally high-order polynomial phase," Appl. Opt. **49**, 1075–1079 (2010).
31. G. Rodríguez-Zurita, A. García-Arellano, N. I. Toto-Arellano, V. H. Flores-Muñoz, R. Pastrana-Sánchez, C. Robledo-Sánchez, O. Martínez-Bravo, N. Vásquez-Pasmino, and C. Costa-Vera, "One-shot phase stepping with a pulsed laser and modulation of polarization: application to speckle interferometry," Opt. Express **23**, 23414–23427 (2015).