Fringe Demodulation using Simulated Annealing with Independent Window Partition.

Iosvani Moré and Francisco J. Cuevas

Centro de Investigaciones en Ópitca, León, Guanajuato, México.

imore@cio.mx, fjcuevas@cio.mx

CENTRO DE INVESTIGACIONES EN OPTICA, A.C.

1. Abstract

A method for the partition of an interferogram used in the window fringe demodulation (WFPD) technique is presented. This algorithm can autonomously divide an interferogram based on the maximum number of fringes desired in each window. Basically, this consists in obtaining the minimum number of sub images consistent with the number of fringes allowed. Each sub-image will serve as an input to the Simulated Annealing algorithm, which estimates the phase map from a parametric function, and its parameters are obtained by means of an optimization process.

2. Introduction

In Optical Metrology, a fringe pattern can be considered as a fluctuation of a sinusoidal signal in bidimensional space, which is related to the physical quantity being measured. The mathematical model that characterizes a pattern of fringes is given by intensity $I(x, y)$, which can be represented through its cosine profile as [1]:

$$I(x, y) = a(x, y) + b(x, y)cos(\phi(x, y) + n(x, y),$$

where $a(x, y)$ represents background lighting, $b(x, y)$ refers to contrast or modulation of the signal, related to the reflectance of the object, $\phi(x, y)$ symbolizes the phase term, and $n(x, y)$ represents high-frequency noise.

3. Simulated Annealing (SA)

The SA technique was formulated by Kirkpatrick, Gelatt and Vecchi in 1983 [2], based on the Monte Carlo method, and it was used in this work to estimate the phase map of each sub-image [3].

Algorithm 1 Simulated Anealing Algorithm

procedure SIMULATED ANEALING
  Input:
  $S_0 \rightarrow$ *Initial solution*
  $T(i) \rightarrow$ *Function of temperature*
  nrep $\rightarrow$ *Number of neighboring solutions*
  while STOP condition **do**
    $S \in Neighborhood(S)$
    $\Delta f = f(S) - f(S_0)$
    for $i : nrep$ **do**
      if $\Delta f \leq 0$ **then**
        $S_0 \leftarrow S$
      else
        $u = Rand(0, 1)$
        if $u \leq e^{-\frac{\Delta f}{T(i)}}$ **then**
          $S_0 \leftarrow S$
        end if
      end if
    end for
  end while
end procedure

4. Interferogram partition in independent windows (IPA)

This work presents a method to divide a pattern of closed and under-sampled fringes into image windows or sub-images [4, 5]. The modulating phase in each sub-image is fitted by a parametric analytic function using the Simulated Annealing algorithm (SA) [2].

In the proposed method, a fringe pattern is partitioned into a small window containing a defined number of fringes. This process consists in obtaining the minimum number of sub-images consistent with the number of allowed fringes. IPA is a recursive method which verifies that each sheet complies with the number of fringes restriction. For this purpose, a tree-type data structure was implemented [6], where each leaf of the tree represents a partition of the interferogram. The algorithm is based on the following steps:

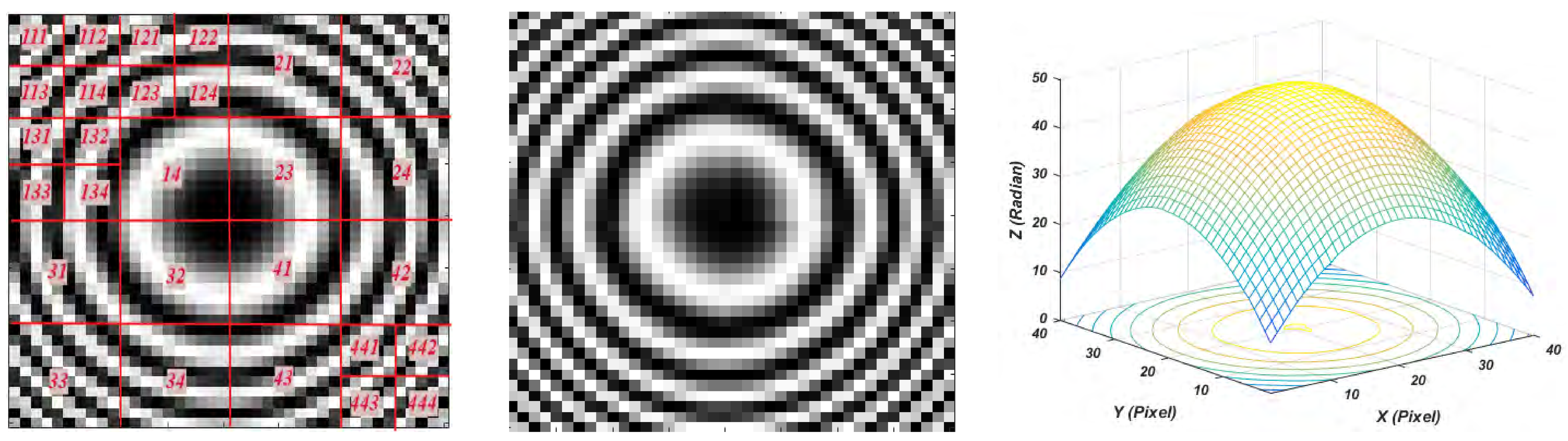
1. A query is performed to verify if the node has a child, and if so, the recursive method with all the children is called again.
2. Otherwise, the node is a leaf, and it is verified whether it complies with the number of fringes restriction. If it does, the process for that window is stopped; otherwise, the interferogram is partitioned into 4 sub-images.
3. The process is repeated until all sheets have a maximum number of required fringes.

5. Experimental Results

The algorithm is tested using a computer generated interferogram in which the mathematical form of the original phase is given by:

$$f(x, y) = 0.054978y^2 + 0.0549781x^2 - 2.199115y - 2.1991150x + 0.2199115, \ x, y \in [0, 40].$$

IPA was applied using a maximum number of fringes equal to 3. For each window, an instance of an RS algorithm was executed to demodulate each image segment. The results of the partition algorithm and the demodulation process are shown in the following figure.



In order to define the quality of the solutions, the mean error between the encountered phase map and the actual phase map is calculated as:

$$Error = \frac{1}{RC} \sum_x \sum_y \left| \frac{f(x, y) - \phi(x, y)}{max(\phi(x, y)) - min(\phi(x, y))} \right| \times 100,$$

where $f(x, y)$ denotes the original shape of the object, R and C represent the number of rows and columns of the phase, and $\phi(x, y)$ is the phase map recovered by the method. Several tests were performed with different configurations of the parameters used during the optimization process, and the best execution yielded an error of 0.3260% between the original and the recovered phase maps.

6. Conclusions

A new technique for demodulating a single interferogram using SA is presented. This model establishes a new model to partition the interferogram automatically, obtaining a number of bands desired in each sub-image, and opens up the possibility to use prior knowledge of the shape of the object and improve the adapted parametric function, rather than use a polynomial function. A computer simulation was used to approximate the phase by means of polynomial functions.

7. References

[1] Daniel Malacara, Manuel Servín, and Zacarias Malacara. *Interferogram Analysis For Optical Testing*. 2005.

[2] S Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *American Association for the Advancement of Science*, 220(4598):671–680, 1983.

[3] Ulises Rodriguez, Miguel Mora-González, Jesus Muñoz Maciel, and Tania Ramirez-delreal. Fsd-hso optimization algorithm for closed fringes interferogram demodulation. *Mathematical Problems in Engineering*, 2016:1–11, 03 2016.

[4] L. E. Toledo and F. J. Cuevas. Optical metrology by fringe processing on independent windows using a genetic algorithm. *Experimental Mechanics*, 48(4):559–569, 2008.

[5] F. J. Cuevas, F. Mendoza, M. Servin, and J. H. Sossa-Azuela. Window fringe pattern demodulation by multi-functional fitting using a genetic algorithm. *Optics Communications*, 261(2):231–239, 2006.

[6] M.T. Goodrich, R. Tamassia, and D.M. Mount. *Data Structures and Algorithms in C++*. Wiley, 2004.