Genetic Learning Based Texture Surface Inspection

H. Zheng[1], S. Nahavandi[1], L. Pan[2] and Y. Xiang[1]
[1]School of Engineering and Technology, Deakin University, Geelong, VIC 3217, Australia
[2]School of Remote Sensing and Information Engineering, Wuhan University
129 Luoyu Road, Wuhan 430079, P.R.China
{hzheng, nahavand, panli,yxiang}@deakin.edu.au

Abstract

This paper presents a novel approach of visual inspection for texture surface defects. It is based on the measure of texture energy acquired by a kind of high performance 2D detection mask, which is learned by genetic algorithms. Experimental results of texture defect inspection on textile images are presented to illustrate the merit and feasibility of the proposed method.

1. Introduction

Surface inspection is usually a bottleneck in many production processes. There is a great number of manufacturing processed where inspection for surface finishing or surface defects is attempted such as textile [1], wooden slabs [2], paper [3] and steel surfaces [4][5]. The most difficult task of inspection is that of inspecting of visual appearance. The visual inspection in most manufacturing processes mainly depends on human inspectors whose performance is generally inadequate and variable. The human visual system is adapted to perform in a world of variety and change; the visual inspection processes, on the other hand, requires observing the same type of image repeatedly to detect anomalies. The accuracy of human visual inspection declines with dull, endlessly routine jobs. Slow, expensive, erratic inspection is the result. Computer based visual inspection is obviously the alternative to the human inspector.

This paper is concerned with the problem of computer inspection of texture surface. In past years numerous approaches have been developed for texture inspection tasks [6]. Previous methods can be divided into two main categories: statistical and structural. The structural approach assumes the texture is characterized by some primitives following a placement rule. In this view, to describe a texture one needs to describe both the primitives and the placement rule. However, the approach is restricted by the complication encountered in determining the primitives and the placement rules that operate on these primitives. Therefore, textures suitable for structural analysis have been confined to quite regular textures rather than more natural textures in practice. In the statistical approach, texture is regarded as a sample from a probability distribution on the image space and defined by a stochastic model or characterized by a set of statistical features. The models which have been used to generate and represent textures include: time series models, fractals, random mosaic models, syntactic models and linear models. The most common features used in practice are based on the tonal properties and the pattern properties. These are measured from first-order and second-order statistics and have been widely used as discriminators between textures.

A major problem with the application of texture inspection to real problems is that textures in the real world are often not uniform, due to changes in orientation, scale or other visual appearance. How to extract robust texture features has become a key issue in the field of texture inspection. In recent years, some researches presented some approaches based on Gabor filters [7][8][9], autoregressive random field model [10] and local binary patterns [11] to extract texture features which are not invariant to changes in rotation or scale. Unfortunately, the degree of computational complexity of these proposed texture measurement is high. In order to solve the problem, this paper presents a new method to extract a robust texture defect feature by genetic algorithms. In his method, the principle texture statistic utilized to represent the texture feature is the normalized "texture energy" derived from Law's approach: the standard deviation of pixel gray scale within a predetermined window size computed after convolution with a optimal texture filter through task-aimed training based on genetic algorithms (GA). The details about the algorithm design are discussed in this paper.

This paper is organized as follows: Section 2 describes the learning algorithm for texture defect detection. Section 3 shows the experimental results and conclusions are given in Section 4.

2. Methodology

2.1 Overview of genetic algorithms

GA is a heuristic search technique for obtaining the best possible solution in a vast solution space. It employs mechanisms analogous to those involved in natural selection to conduct a search through a given parameter space for the maximum/minimum of some objective function. To apply a GA, an initial population is generated and the fitness of each member of the population is evaluated. The algorithm then iterates the following: members from the population are selected for reproduction in accordance to their fitness evaluations. The reproduction operator are then applied, which generally include a crossover operator that models the exchange of genetic material between the parent chromosomes and a mutation operator to maintain diversity and introduce new alleles into the generation, or a combination of both, to generate the offspring of the next generation. The fitness of the offspring is then evaluated, and the algorithm starts a new iteration. The algorithm stops when either a sufficiently good solution is found, or after a predetermined number of iterations.

GA has been successfully applied in numerous commercial and industrial fields. For image processing problems, some recent attempts in image segmentation, primitive extraction, scene recognition and image interpretation are reported in the literature [12][13][14][15]. In this study, we apply GA to obtaining optimal filter parameters and segmentation threshold for texture defect detection.

2.2 Genetic algorithms for learning defect detection

Basically, a genetic algorithm includes six issues such as encoding scheme, evaluation, selection, crossover, mutation and stopping criterion. In this study several issues must be considered as follows.

2.2.1 Encoding scheme

Now we define a set of individuals in a population generated during t generation cycles. $P(t)=\{I_k | k=1,2,\ldots,n\}$, where n is the number of individuals or the population size. The size affects both the ultimate performance and the efficiency of GA. Each individual is generated by some encoded form known as a chromosome. In this study, a segmentation threshold and parameters of a 5 by 5 filter are encoded a float chromosome. Fig. 1 describes the encoding scheme.

2.2.2 Evaluation function

We need to define a function which measures the detection quality of a chromosome. The evaluation function is defined as below:

$$f = \frac{M}{N} \times 100\,\% \qquad (1)$$

where M is the number of the samples detected correctly, N is the total number of training samples. The greater the value of f, the higher the chromosome's fitness.

For each chromosome, the procedure to recognize defects consists of following three steps:

1) Decode a chromosome and get a filter and a threshold.
2) Convolve the all training images by the filter. The 2D convolution of the image I(i,j) and filter A(i,j) with size 2a+1 by 2 a +1 is given by the relation

$$F(i,j) = A(i,j) * I(i,j) = \sum_{k=-a}^{a} \sum_{l=-a}^{a} A(k,l) I(i+k, j+l) \qquad (2)$$

For a 5 by 5 filter, a is 2.

3) Calculate the standard deviation values of the convolved training images. The images whose standard deviation values are greater than the threshold are recognized as defective textures, otherwise, they are recognized as non-defective textures.

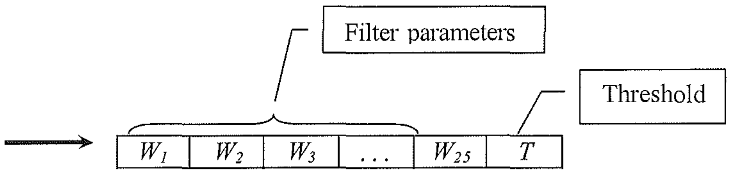
2.2.3 Genetic operators

For GA, the two operations, namely, crossover and mutation, will be implemented. In this study, a single-point crossover is employed. For the single-point crossover , the crossover-point position in a string are randomly selected. Mutation is carried out by performing a random replacement operation on some randomly selected position of the parent strings.

The parameters that control the GA can significantly

W_1	W_2	W_3	W_4	W_5
W_6	W_7	W_8	W_9	W_{10}
W_{11}	W_{12}	W_{13}	W_{14}	W_{15}
W_{16}	W_{17}	W_{18}	W_{19}	W_{20}
W_{21}	W_{22}	W_{23}	W_{24}	W_{25}

(a) A 5 by 5 filter architecture . $W_i \in [-2,2], i = 1,2,\ldots,25$



(b) A chromosome architecture. $T \in [0,512]$

Figure 1. Encoding scheme

affect its performance. The most important parameters are the crossover rate and the mutation probability. In this paper, we introduce a dynamic adaptive setting method. The crossover rate P_c and the mutation probability P_m are given as follows

$$P_c = \frac{f_{\max} - f'}{f_{\max} - f_{\min}} \quad (3)$$

$$P_m = \frac{f_{\max} - f}{f_{\max} - f_{\min}} \quad (4)$$

where f is the fitness value of currently evaluated chromosome, f_{max} is the maximum fitness value of the population, f_{min} is the minimum fitness value of the population, and f' is larger of the fitness values of the parent chromosomes to be crossed.

From Eq. 3 and Eq. 4, we can see that P_c and P_m decrease in accordance to the increase of the fitness value of evaluated chromosome. For the best chromosome of the population, P_c =0, and P_m=0. It is helpful to preserve 'good' chromosomes of the population. In addition, each chromosome except best chromosome is subjected to the crossover and mutation operations. It leads to the diversity of the population and speeds up the procedure of searching for the best solution.

2.2.4 Stopping criteria

The genetic algorithm will be iteratively performed on training image samples until a stopping criterion is satisfied. The stopping criterion is either the percentage of the fitness function value improvement between two consecutive iteration is smaller than a threshold or the number of iteration is greater than a given threshold. In this study, the maximum number of iterations of GA is 100.

2.2.5 The procedure of proposed genetic learning algorithm

The procedure of learning thresholds and defect filters is given below, where P(t) is a population of candidate solutions at generation t.

t=0;
initialize P(t)
evaluate P(t)
while not (termination condition)

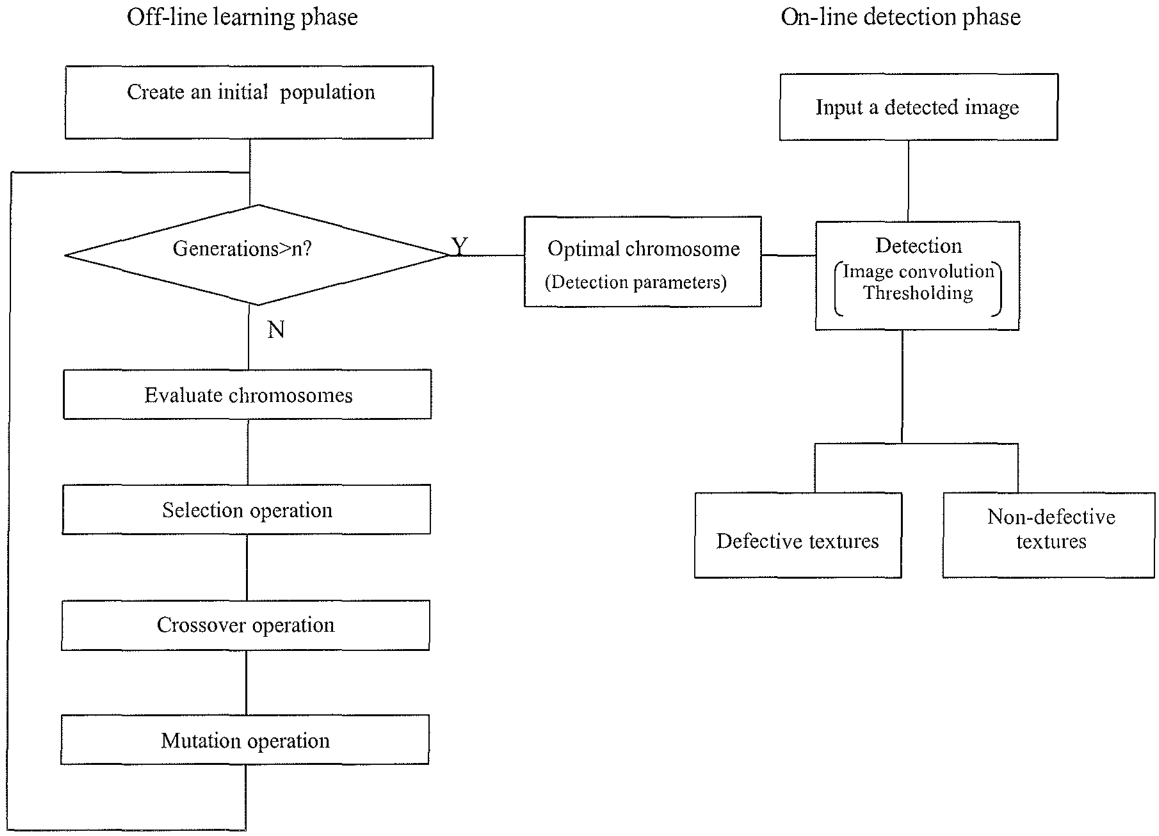


Figure 2. Flowchart of the defect detection based on genetic learning

```
  begin
   t=t+1;
   reproduce P(t) from P(t-1)
   recombine P(t) by crossover and mutation operator
   evaluate P(t)
  end;
end while
```

2.3 Defect detection procedure based on learned parameters

After learning procedure, the acquired optimal solution (the texture object filter and threshold) is used to detect defects.

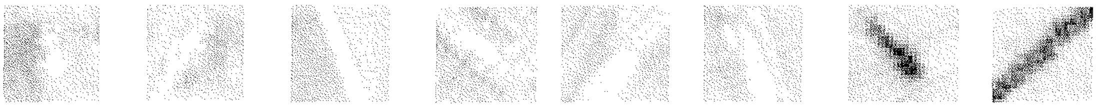
Firstly, according to Eq.2, convolve the testing image by the learned filter. Secondly, calculate the standard deviation within a 19 by 19 window at each point. The standard deviation is defined as the texture feature (TE) at the point. Thirdly, compare the TE value at each point with the learned threshold. If the TE is greater than the threshold, the point belongs to the defect region. Otherwise, it belongs to the defect-free region. Finally, a post-processing based on morphology dilation and erosion processing is employed to remove noise.

Figure 2 shows the whole flowchart of defect detection based on the genetic learning.

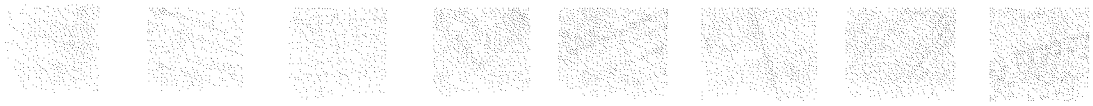
3. Experimental results

In this section we present experimental results on textile images. The training textile images used in the experiment are from the TILDA textile image database created at the University of Freiburg, Germany. Some of them are shown in Figure 3 and Figure 4. In TILDA, textile samples are grouped as defect-free or having a certain type of defect. For each defect type, defect-free samples are provided. In our experiment, we learned a defect filter and segment threshold for each texture type. For each texture type, a total of 30 sample images are selected for learning defect detection. These images are sampled from the same type of textile images, and they are divided into two texture classes. One of them belongs to defect-free texture samples, the other belongs to defect texture samples. Figure 3(a) and 4(a) show some defect texture samples under different orientations, scales and shapes. Figure 3(b) and 4(b) show some defect-free texture samples under different orientations, scales and shapes.

The experiment of texture defect detection is performed in two stages. The first stage is the training phase where adaptive filters and segmentation thresholds are obtained through learning the training texture samples using GA. The second stage is to detect defects using these filters and thresholds. Figure 5(b) and 5(d)

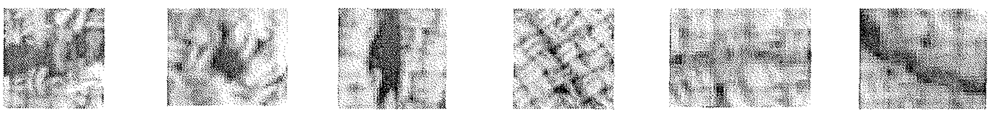


(a) Defect texture sample images

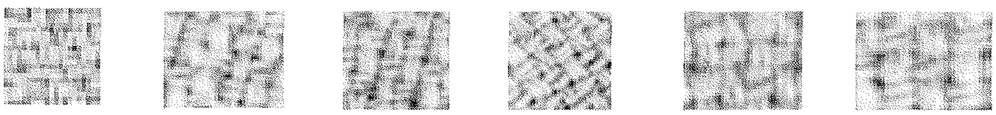


(b) Defect-free texture sample images

Figure 3. Textile sample image set A



(a) Defect texture sample images



(b) Defect-free texture sample images

Figure 4. Textile sample image set B

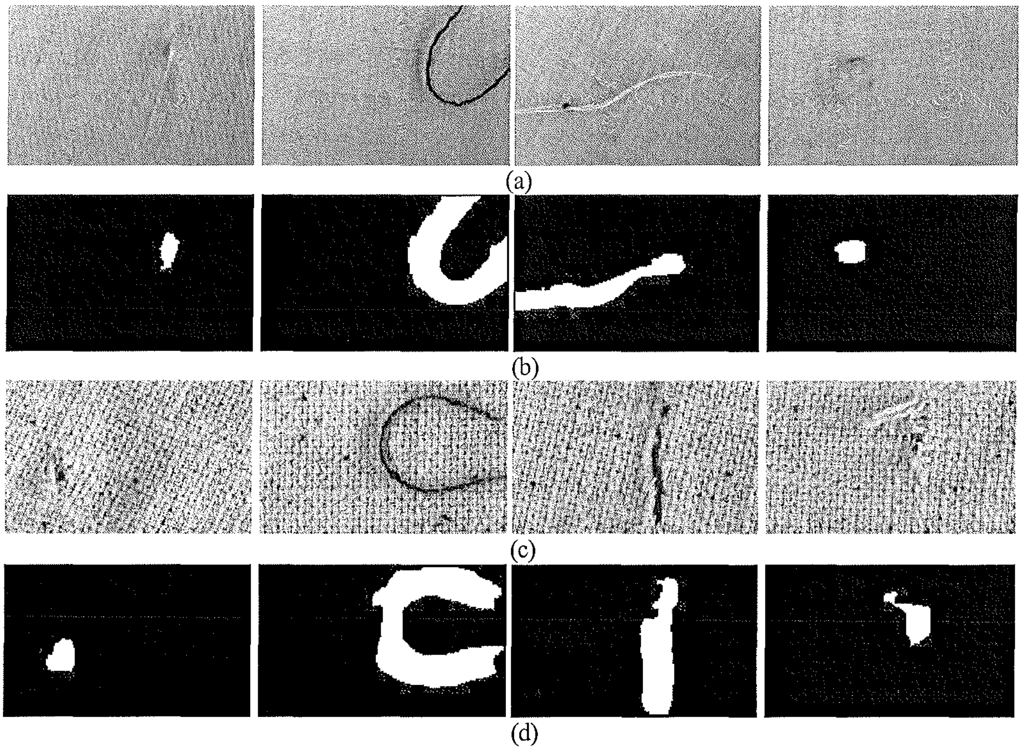


Figure 5. Four examples of textile inspection by proposed algorithms.
(a) (c) Original textile images.
(b) (d) Defect detection results.

show the results of extracting defects from images shown in Fig. 5(a) and 5(c), respectively. From Fig.5, it can be seen that, although defect appearances on different textile surfaces are much different, they can be extracted correctly by proposed method. We test our algorithms on 100 textile images. The correct rate can reach 91.2%.

4. Conclusions

This paper presents the approach for texture inspection based on genetic learning and demonstrates its ability for texture inspection. The method can extract defects with different sizes, shapes and orientations, and its simple computational form is very suitable for hardware implementing. Although this paper has been devoted almost entirely to the textile inspection problem, the principle of proposed approach can be applied to other machine vision applications.

References

[1] C. Neubauer, Segmentation of Defects in Textile Fabric. In: *Proc. Int'l Conf. on Pattern Recognition*, pp.688-691, 1992

[2] C.W. Kim and A.J. Koivo, Hierachical Classification of Surface Defects on Dusty Wood Boards. *Pattern Recognition Letters*, Vol.15, pp.713-721, 1994.

[3] J. Iivarinen and A. Visa. An Adaptive Texture and Shape Based Defect Classification. In: *Proc. Int'l. Conf. On Pattern Recognition*, pp.117–123, 1998.

[4] D. Brzakovie and N. Vujoivie, Designing a Defect Classification System : A Case Study, *Pattern Recognition*, Vol.29, No. 8, pp.1401-141, 1996.

[5] T.S. Newman and A.K. Jain. A Survey of Automated Visual Inspection. *Computer Vision and Image Understanding*, Vol. 67, pp.231–262, 1995.

[6] K. Y. Song, M. Petrou, and J. Kittler. Texture Defect Detection: A Review. In *Proc. SPIE Applications of Artificial Intelligence X: Machine Vision and Robotics*, Vol.1708, pp.99-106, 1992.

[7] D. Dunn and E. Higgins. Optimal Gabor Filters for Texture Segmentation. *IEEE Transactions on Image Processing*, Vol.4, No.7, pp.947–964, 1995.

[8] A. Teuner, O. Pichler, and B. Hosticka, Unsupervised Texture Segmentation of Images using Tuned

Matched Gabor Filters. *IEEE Transactions on Image Processing*, Vol.4, No.6, pp.863–870, 1995.

[9] T. Weldon, W. Higgins, and D. Dunn, Efficient Gabor Filter Desing for Texture Segmentation. *Pattern Recognition*, 29(12) (1996) 2005–2015.

[10]M. Pietikäinen , T. Ojala and Z. Xu, Rotation-Invariant Texture Classification Using Feature Distributions. *Pattern Recognition*, Vol.33, pp.43-52, 2000.

[11]T. Ojala, M. Pietikäinen and T. Mäenpää , A generalized Local Binary Pattern Operator for Multi-resolution Gray Scale and Rotation Invariant Texture Classification. In: *Proc. 2nd Int'l. Conf. on Advances in Pattern Recognition*, Rio de Janeiro, Brazil, pp.397-406, 2001.

[12]B. Bhanu, S. Lee and J. Ming, Adaptive Image Segmentation using a Genetic Algorithm, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol.25, No.12, pp.1543-1567, 1995.

[13]G. Roth and M. D. Levine, A Genetic Algorithm for Primitive Extraction, In: *Proc. 4th Intl. Conf. Genetic Algorithm*, San Diego, pp.487-497, 1991.

[14]D. Prabhu, B.P. Buckles, and F.E. Petry, Genetic Algorithms for Scene Interpretation from Prototypical Semantic Description , *International Journal of Intelligent Systems*, Vol.15, No.10, 901-918, 1998.

[15]A. Hill and C. J. Taylor, Model-based Image Interpretation using Genetic Algorithms, *Image and Vision Computing*, Vol.10, No.5, pp.295-300, 1992