ELSEVIER

Image and Vision Computing 18 (2000) 951–957

image AND vision COMPUTING

www.elsevier.com/locate/imavis

Industrial inspection using Gaussian functions in a colour space

L. Bergasa[1,a,*], N. Duffy[2,b], G. Lacey[2,b], M. Mazo[1,a]

[a]*Departamento de Electronica, Universidad de Alcala, Escuela Politencnica, Alcala de Henares, 28805 Madrid, Spain*
[b]*Computer Vision and Robotics Group, Trinity College, Dublin, Ireland*

Received 15 December 1999; received in revised form 15 December 1999; accepted 3 February 2000

Abstract

This paper presents an original method of modelling the colour distributions of images using 2D Gaussian functions and its application to flaw detection in industrial inspection. 2D Gaussian functions are used to model the colours that appear in the non-flawed images in an unsupervised manner. Pixels under test are compared to the colour distribution from training images. 140 images have been tested and the results are given. This method has a wide range of applications for detecting colour separable objects in images. It also has great potential in industrial inspection due to its speed, accuracy and unsupervised training.

Keywords: Colour object detection; Gaussian functions; Colour clustering; Competitive learning histogram; Automated industrial inspection

1. Introduction

This paper describes the use of Gaussian functions in a 2D colour space to perform industrial inspection on moulded plastic filter casings. The filter casings are manufactured in a variety of different colours and structural designs. They are transparent and have a highly complex structure, causing many internal shadows. The casings need to be inspected for small burn marks and discoloration. The flaws vary not only in size and location but also in colour. The colours present in a burn cover a range that includes yellow, green, brown, and black to blue. The parts are small (approx. 3 cm in diameter) and complicated and, because of this, manual inspection by humans requires a high level of concentration. In fact, human inspection has proven to be very inaccurate; only during the first half-hour are humans able to perform significantly better than random. Even though 6–8 different humans inspect each filter casing, many flawed casings are not discovered, which has created the need for an automated inspection system.

The casings are manufactured at the rate of 1.4 per second. Owing to the complexity and real time requirements, established vision techniques, which search for structural defects, are not applicable. Instead of focusing on the more difficult question of checking the structure, the approach taken in this paper is to check that the colours present in an image fall within modelled colours. Pixels whose colours do not fall into these modelled colours are considered flaws. It is not necessary that the flaws themselves have uniform colour, because the Gaussian functions model the non-flaw colours, not the flaw colours. If the light and the background conditions do not change then it can be assumed that the untrained colours will be flaws. In testing it has been shown that the program is tolerant to small variations in the lighting conditions. However, the best results are achieved when the background is controlled as much as possible. For example a dust particle in the background will be detected as a flaw. In the application for industrial inspection it is not a problem because this system works in a clean environment (Fig. 1).

The use of colour information in image processing is becoming more and more common place as the availability of cheaper technology increases. There have been several efforts to use colour to segment images. In Ref. [1] a second order basic function is introduced to pick up the pixels matching the specified colour. Statistical classification is a standard method largely employed in which multivariable Gaussian functions are fitted to the different colours in the image [2–4]. The colour profile method [5,6] is constructed by comparing the colour histograms of the images of flawed components to those of the non-flawed components. All these methods need supervised learning. In recent years some unsupervised segmentation methods based on

* Corresponding author.
E-mail addresses: bergasa@depeca.alcala.es (L. Bergasa); nikla.duffy@cs.tcd.ie (N. Duffy); gerard.lacey@cs.tcd.ie (G. Lacey); mazo@depeca.alcala.es (M. Mazo).

[1] http://www.depeca.alcala.es.

[2] http://www.cs.tcd.ie/research_groups/cvrg/.

0262-8856/00/$ - see front matter
PII: S0262-8856(00)00035-4

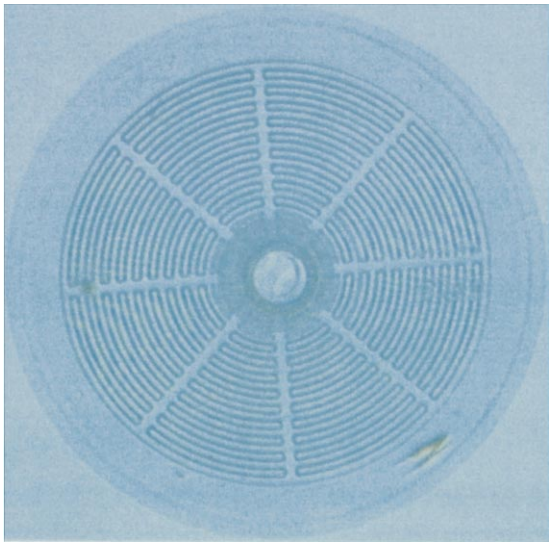

Fig. 1. Clear filter casing with flaw.

competitive learning have been developed. In Ref. [7] a self-organising map (SOM) is used for the colour segmentation phase and a multi-layer perceptron is trained to label the regions produced by the segmentation process in a supervised manner. In Ref. [8] a SOM method is applied to the segmentation process and a *k*-means algorithm for the labelling.

The main contribution of this work is the use of a Gaussian model to approximate the histogram of a colour image in an unsupervised way. The system has two phases: learning and testing. In the first phase the different colours of a non-flawed image are learnt from a set of non-flawed training images. A Gaussian function is used to model each main colour [3]. The Gaussian model of the colours appearing in the non-flawed images can be used to classify pixels whose colours fall outside the Gaussian functions as flaw pixels. The model is constructed in two stages: first, the number of Gaussian functions needed to model the non-flawed images is found using an approximate colour histogram; second, the parameters of the Gaussian functions (mean and covariance) are calculated using a competitive learning strategy. To test an image for the presence of a flaw each image pixel is assigned a membership value and a threshold is applied to this value in order to classify the pixel as flawed or non-flawed.

2. Learning phase

The system learns the different colours of the non-flawed images in a non-supervised manner. It is presented with a set of non-flawed training images. The training images are analysed and their colours approximated with a set of Gaussian functions. The RGB space is used because the raw output format of the frame grabber is RGB and therefore a colour space transformation was not needed. Practical experiments demonstrated that the blue band in the RGB space does not have a significant effect during the learning phase because CCD cameras are less sensitive to blue relative to red and green. On the other hand, the lighting used does not have a high blue content relative to red and green. This is why it was decided to eliminate the blue component and work in the RG colour space. As light conditions were quite stable, it was demonstrated that illumination normalisation was not necessary. A white balancing in the process initialisation performs correctly enough to make the system independent of illumination conditions.

To demonstrate this technique, a specially constructed image was created by patching together four sections from real images, and since there were no flaws present in any of the image areas, flaw pixels from other images have also been added. Fig. 2(a) shows this image, which, although artificial, is interesting because it has four clear peaks in the RG histogram space, which is shown in Fig. 2(b). Fig.

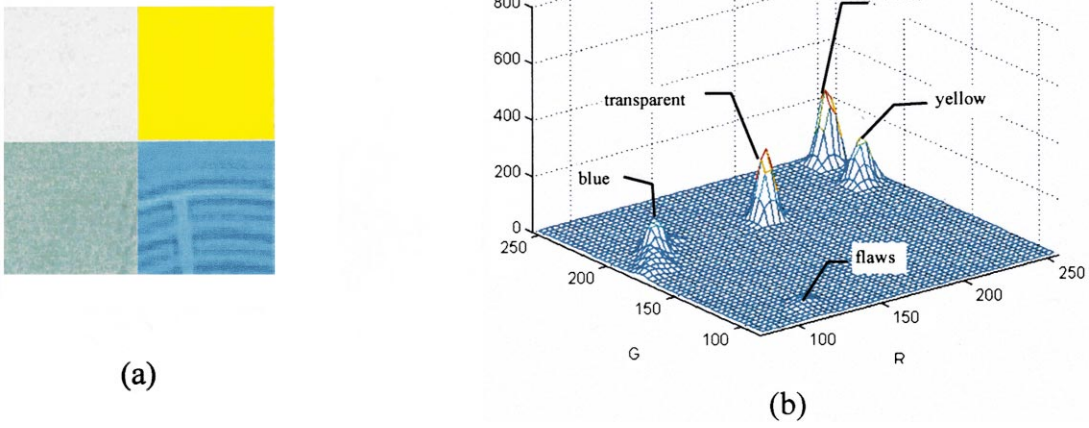

Fig. 2. (a) Image areas, (b) the *RG* histogram.

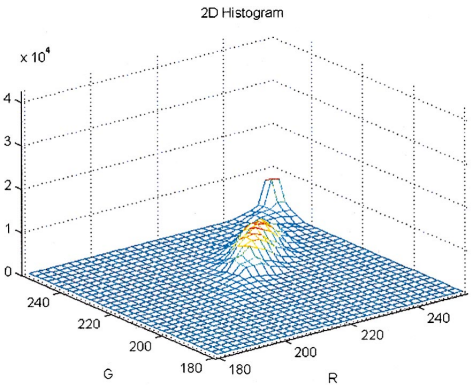

Fig. 3. *RG* histogram of a non-flawed image.

2(b) shows that each colour has a form very similar to a Gaussian function in the RG space. Indeed these distributions can be approximated using four Gaussian functions and it is easy to see that the flaw colours are separable.

The RG histogram of a real transparent non-flawed image can be seen in Fig. 3. It is not as easy to see the Gaussian form of this histogram as it was to see those in Fig. 2(b). The Gaussian form is clearer in the 1D projections of the histogram onto the *R* and *G* axis, as in Fig. 4. The image of a casting, like that of the image in Fig. 1, will have two main components: the plastic filter and the background. The background colours can be modelled with a Gaussian near white in the colour histogram having a small covariance. The filter casing colours can be modelled with Gaussian functions having larger covariances. This is because the colours of the filter casing vary more than the colours of the background.

An automatic method is required to calculate the Gaussian approximation of the histogram of a non-flawed image. To achieve this we use a two stage clustering method: in the first stage, the number of Gaussian functions needed to model the colours present is calculated; and in the second stage a competitive learning strategy calculates the parameters of the Gaussian functions.

2.1. Calculating the number of Gaussian functions in the model

The main colours in an *M* by *M* image are found using a colour histogram of the image. The user can choose the resolution of the histogram by specifying the number of bins, *N*, along each axis. Each colour axis is divided up into *N* intervals of equal size, *S*, equal to the dynamic range of the colour axis, *P*, divided by the number of bins, *N*. The histogram will be an $N \times N$ matrix and every bin will be initialised with zero. For every pixel, $x = (x_R, x_G)$, of the image

$$H(f_{\text{bin}}(x)) = H(f_{\text{bin}}(x)) + 1 \tag{1}$$

where

$$f_{\text{bin}}(x) = \left(truc\left(\frac{x_R}{S}\right), truc\left(\frac{x_G}{S}\right)\right) \tag{2}$$

The structure of the bins in a colour histogram implies an equality between the colours. If the histogram has a bin size, $S = P/N$, then it is possible that colours less separated than *S* may fall into the same colour bin and will therefore be considered the same colour. In this application only the main colours in the image are sought and a 10×10 histogram has proven optimal for the task.

The number of Gaussian functions to use in the model is calculated by thresholding the bins in the histogram. In this application there is a very big bin that contains the background colour and a big bin that contains the main colour of the plastic filter. There are also some small bins that represent marginal colours in the filter and/or in the background. Fig. 5 shows the approximate histogram for the images of Figs. 2 and 3. The number of Gaussian functions depends on the threshold used on the histogram bins. The effect of the value of the threshold is discussed in Section 4.

This system localises the bins bigger than the threshold in the histogram and uses the bin position in the colour space to estimate the positions of the mean of the Gaussian functions. Of course these positions are not exact but they are a good

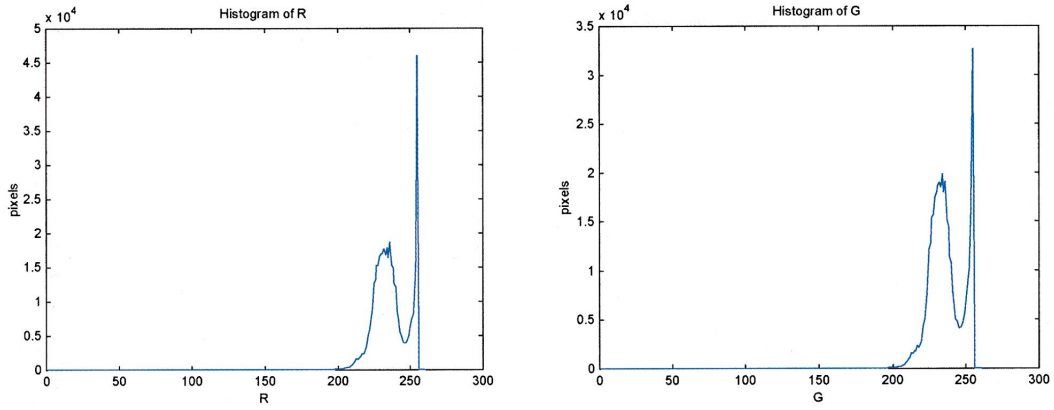

Fig. 4. The *R* and *G* projections of the 2D histogram.

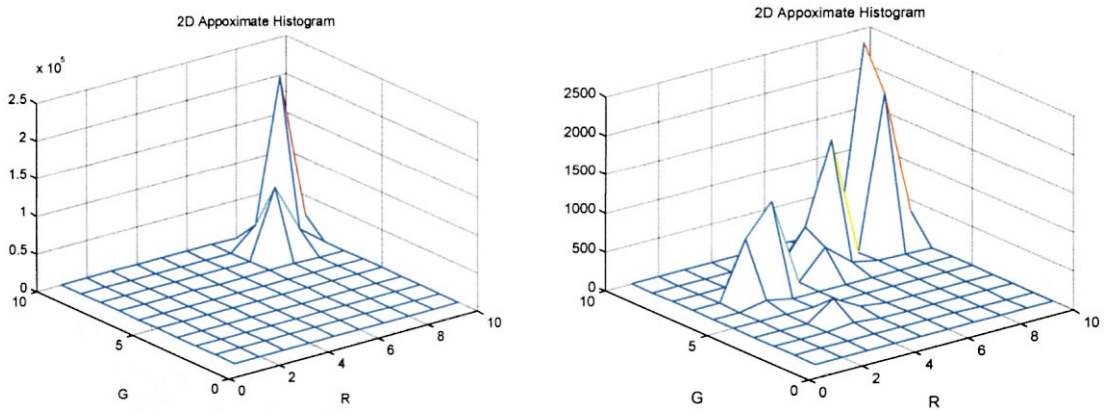

Fig. 5. Approximate histograms (left Fig. 3 and right Fig. 2).

approximation and have a maximum error of $\pm P/2N$. The exact means are calculated in a second stage.

2.2. Competitive learning

In order to adjust the parameters of the Gaussian functions a competitive algorithm proposed by Kohonen called vector quantization, VQ [9] is used. A parameter vector approximates a large amount of information. In this case the information is the RG colour values of the pixels and the vector contains the position of the centres of the Gaussian functions. To apply VQ the number of Gaussian functions and an approximate starting position for each must be known, which is the case. The initial position parameter is very important in this method because it can greatly influence the accuracy of a final result. See Refs. [10,11] for a discussion on this subject. In this case the algorithm starts with a good estimate of the final position and this greatly increases its ability to converge on the true values.

The colours of the pixels of an image are defined as follows:

$$X = (x_1, x_2, \ldots, x_n);\quad n = M \times M;\quad x_i = (x_{iR}, x_{iG}) \tag{3}$$

and the c Gaussian functions in the positions defined by the v_i vectors are defined as follows:

$$V = (v_1, v_2, \ldots, v_c);\quad v_i = (v_{iR}, v_{iG}) \tag{4}$$

The VQ method approximates the density functions of likelihood, $f(x)$, of a stochastic variable $x \in R^2$ using a finite number of vectors V, called nets. Two layers are considered: the input layer and the competitive layer as seen in Fig. 6.

In order to calculate the best approximation of X Kohonen defines a mean square error function shown in Eq. (5). The minimum of this function is found by applying the gradient descent method. A recurrent equation, like Eq. (6), is used for moving the nets in the colour space.

$$E = \int \|x - v_i\|^2 f(x)\, \mathrm{d}x \tag{5}$$

$$v_i(t+1) = v_i(t) + \alpha(t)[x(t) - v_i(t)] \tag{6}$$

The Euclidean distance between each pixel and the vectors is calculated. The closest vector to each pixel is selected and this vector is moved closer to the pixel by a proportional distance to the distance from the pixel. This process is repeated until the vectors are being moved less than a threshold defined by the user. The quantity by which the vectors are moved are controlled by the parameter $\alpha(t)$ and is decreased with the time. Fig. 7 shows the learning algorithm.

Fig. 8 represents the initial positions of the centres of the Gaussian functions (marked with dark circles) and the final positions after the competitive learning (marked with dark

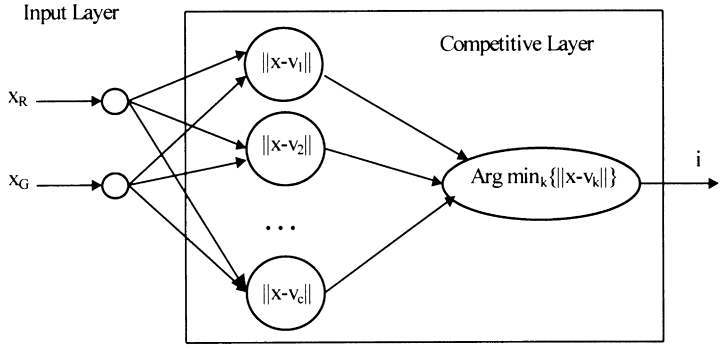

Fig. 6. Competitive learning.

$x \in R^2$:pixel colour, c: number of Gaussian functions, T: maximum number of steps ε :maximum error.

1. Start with $V=(v_{10}, v_{20}, \ldots, v_{c0})$ in the positions calculated by the first stage
2. Start $\alpha(0) \in (0,1)$

For t = 1,2,…,T

For k=1,2,…,n

Find $i=\text{argmin}_r\{\|x_k - v_r\|\}$

Move the winner:

$$v_i(t+1) + v_i(t) + \alpha(t)[x_k(t) - v_i(t)];$$

New k;

end;

Calculate :

$$E_t = V_t - V_{t-1} = \sum_{k=1}^{n}\sum_{r=1}^{c} v_{rk}(t) - v_{rk}(t-1)$$

If $E_t \leq \varepsilon$ stop

else $\alpha(t) = \alpha(1-t/T)$

New t

end:

Fig. 7. The learning algorithm.

crosses) for the histograms of Figs. 2 and 3. For the histogram of Fig. 3 a model with two functions has been chosen.

Now that the correct centres of the Gaussian functions are known, the covariance of each one can be calculated. The image pixels are clustered using Euclidean distance and the centres of the Gaussian functions. The covariance of the sets of pixels assigned to each Gaussian function centre is calculated. This process is repeated with all the images of the non-flaws training set and the mean values of all the parameters is calculated. These are recorded and used to estimate the non-flawed colours in the testing phase.

3. Testing

In the learning phase a mathematical model of the colours of the non-flawed images has been constructed and it can be used to test if the colour value of a pixel belongs to the non-flawed colours or to the flawed colours. This is achieved by calculating the value of all the Gaussian functions for each pixel. Each Gaussian function gives the likelihood that a pixel belongs to that particular learned colour. Eq. (7) is used for calculating the likelihood of a pixel belonging to a colour, where $m_i = (m_{iR}, m_{iG})$ and C_i are given in Eq. (8). For every pixel this has to be calculated for each of the Gaussian functions in the model. If these values are less than a threshold it can be assumed that the colour of that pixel is a flaw.

$$f(x/\text{color}_i) = \frac{1}{2\pi|C_i|^{0.5}}\, e^{-0.5(x-m_i)^T C_i^{-1}(x-m_i)} \tag{7}$$

$$m_{iR} = \frac{1}{M_i}\sum_{j=1}^{M_i} x_{iR}(j) \quad m_{iG} = \frac{1}{M_i}\sum_{j=1}^{M_i} x_{iG}(j)$$

$$C_i = \begin{bmatrix} \sigma_R^2 & \sigma_{RG}^2 \\ \sigma_{GR}^2 & \sigma_G^2 \end{bmatrix} \tag{8}$$

Good detection depends on having the correct threshold. To find the correct threshold a set of non-flawed images is tested and the threshold set so that no flaws are detected. In this application, 10 non-flawed images have been used. Fig. 9 shows the detected flaws for two different thresholds

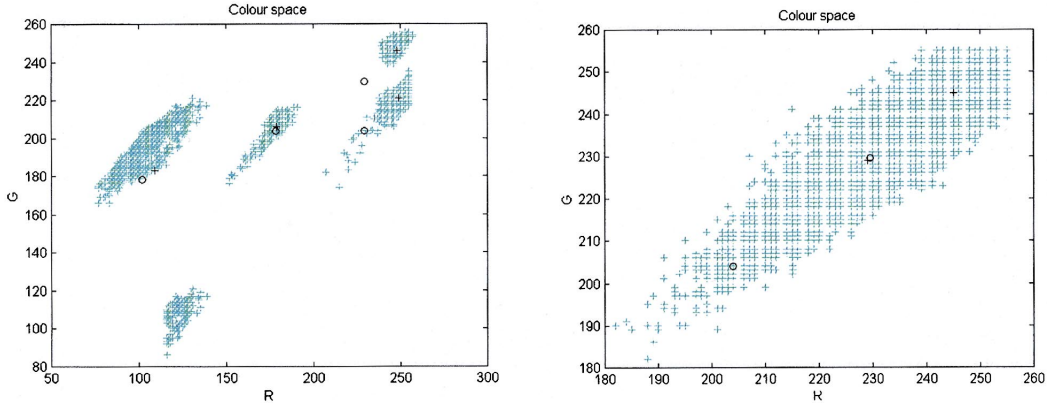

Fig. 8. Position of the centres of Gaussian functions (left Fig. 2 and right Fig. 3).

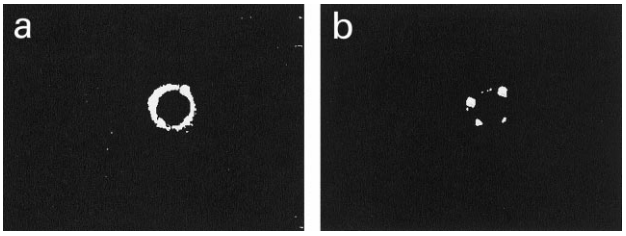

Fig. 9. Flaw detection with different thresholds.

using a model with two Gaussian functions. In image 9(a) a threshold of 0.01 was used and many non-flawed pixels are detected. In image 9(b) the threshold is lowered to 0.0001 and then only true flawed pixels are detected. This is a large enough margin to separate images that contain flaws from those that do not, even though some individual flawed pixels, whose colours are very close in colour to trained colours, are missed.

4. Discussion

It is possible to view the likelihood function of a colour as an approximation of the Mahalanobis distance

$$d_M(x, m_i) = (x - m_i)^{\mathrm{T}} C_i^{-1} (x - m_i) \tag{9}$$

The likelihood of a colour is inversely proportional to the Malalanobis distance from the centre of the Gaussian function and the colour. The colour space can be divided into two areas, flawed and non-flawed colours. In Fig. 10 the border of decision between the flawed and non-flawed areas in function of the number of Gaussian functions in the model is shown.

The learning phase used 10 non-flawed images. The border is in red (or dark grey) and the pixels of the non-flawed images in blue (or light grey). From Fig. 10 it is clear to see that the best estimation of the non-flawed area is that obtained with a model that uses two Gaussian functions. In image (a) the Gaussian cannot converge perfectly on the non-flawed area because there are two main colours and they are being estimated with only one function. In image (b) we see the best approximation for the non-flawed area, because with two Gaussian functions the colours of the image can be estimated very well. If more Gaussian functions are used, such as the four in image (c), the estimation is less accurate. This is because each Gaussian function specialises in a particular colour area of the image and, consequently, the non-flawed areas increased due to lateral colours having a greater effect. Fig. 10 shows that the best estimation is obtained with two Gaussian functions and this makes sense because there are two main colours in the image. With this method the best Gaussian approximation of a colour histogram is done because the optimal number of functions are chosen in each case.

5. Results

With non-optimised code running on a P166 with NT OS the system is able to test a loaded image of 768 × 576 pixels in 1.0 s. The model was tested using the Mahalanobis distance because it is faster than using Gaussian functions. Flaws smaller than 0.2 mm (4 pixels) are very unlikely to be detected by humans, and are therefore not considered real flaws. On the other hand, casings with small flaws also cannot be viewed as non-flawed. To avoid this ambiguity the small flawed casings have been tested in a separate category, small flaws. The small flaws figures are an artefact of the resolution of the system and therefore are not a measure of the technique. As it can be seen in Table 1 different models have been done with one, two and four functions. The best results are obtained using models with two Gaussian functions.

A comparison between the results of the proposed method and the colour profiling one in a RGB colour space [5,6] is also presented in Table 1.

As it can be seen, the results obtained with the Gaussian model method for an optimum number of functions are better than those provided by the colour profiling one, especially in the small flaws, where the conditions of the classification are more complex. It is remarkable that the colour profiling method works in a 3D space (RGB) and the

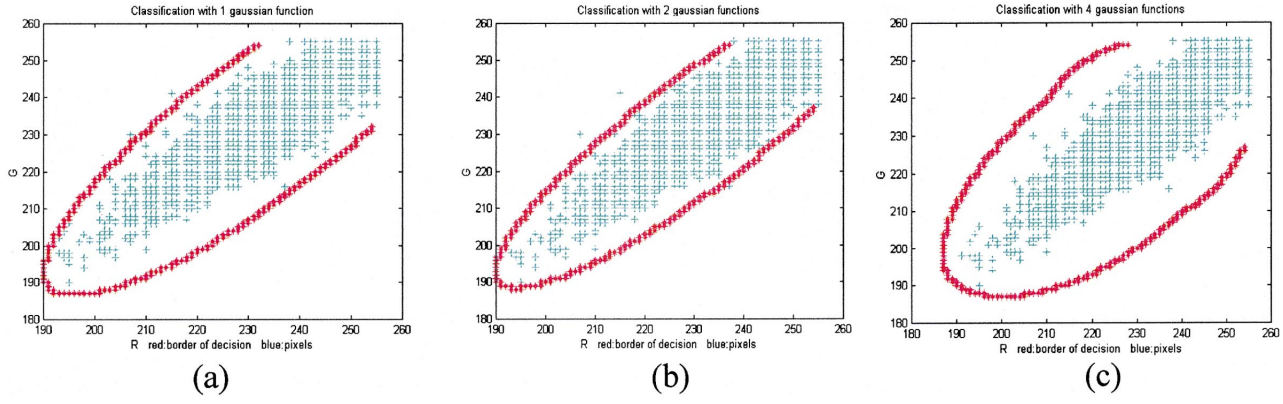

Fig. 10. Mahalanobis distance classification.

Table 1
Results of the system (training set of 10 for Gaussian model and 80 (40/40) for colour profiling)

Clear top filter casings	Gaussian model						Colour profiling	
	RG (1G)		RG (2G)		RG (4G)		RGB	
Correctly classified flawed	58	95%	60	98%	59	97%	57	93%
Misclassified flawed	3	5%	1	2%	2	3%	4	7%
Correctly classified small flawed	22	69%	25	78%	22	69%	10	32%
Misclassified small flawed	10	31%	7	22%	10	31%	22	68%
Correctly classified non-flawed	43	94%	44	98%	44	98%	44	98%
Misclassified non-flawed	3	6%	2	2%	2	2%	2	2%

Gaussian in a 2D (RG). Then, the first method needs a higher training set (40 non-flawed/40 flawed images) than the second one (10 non-flawed images). Finally, it was demonstrated that the colour profiling method is more sensitive to small changes of light conditions than the Gaussian one.

6. Conclusions

This paper has presented a novel colour approximation method that models the colour distribution in an image using Gaussian functions. The positions of the main colours in the image are estimated and these estimates are used to initiate a competitive learning strategy. The improved centres are used to cluster the image and the covariance is calculated giving the two parameters needed for each Gaussian function. Finally, the model is used to test the membership values of pixels and to classify them as flawed or non-flawed pixels. This method has obtained better results than the colour profiling method. It is simpler to train than a colour profile, because training uses only a reduced number of non-flawed images. It uses a mathematical function in the learning stage and, consequently, it is more robust to noise, more exact, and needs less parameters than the colour profiling approach. The method is similar to SOM because it is able to learn the colours of an image without supervision, but better because it is able to know the optimal number of vectors in the image in order to do the approximation. This classification method is better than the *k*-means applied in Ref. [8], because the Mahalanobis distance has been used as the distance metric in the testing phase. The method is also similar to the Bayesian classification in that it is able to obtain the mathematical functions that best approximate the real colours and then use these functions in order to classify the pixels. However, it is an improvement on Bayesian methods and the learning vector quantization because it does not need supervision. In fact the method performs as an optimal non-supervised Bayesian classifier. The method is accurate, real-time and can be trained easily and quickly by the user and it learns in a non-supervised manner, making it ideal for colour segmentation in industrial inspection tasks.

Acknowledgements

I would like to thank the Computer Science Department at Trinity College of Dublin for the opportunity to work in the Computer Vision and Robotics Laboratory and to apply this method to the industrial inspection problem. The authors would like to acknowledge the funding of the Universidad de Alcala, the Enterprise Ireland project ST/96/108 and Millipore Ltd, and they also express their gratitude to The Spanish International Science and Technology Commission (CICYT) for the support given through the project SIAMO (Code TER 1957-C03-01) in the development of the technique applied in this work.

References

[1] Y. Gong, M. Sakauchi, Detection of region matching specified chromatic features, Computer Vision and Image Understanding 61 (2) (1995) 260–269.
[2] E. Littmann, H. Ritter, Adaptive color segmentation—a comparison of neural and statistical methods, IEEE Transactions on Neural Networks 8 (1) (1997).
[3] J. Yang, A. Waibel, Tracking Human Faces in Real-Time, CMU CS. Technical Report, CMU-CS-95-210, November 1995.
[4] F.J. Rodrigez, Contribution a un sistema de detection de bordes de carreteras, mediante vision artificial, orientado al guiado de un robot movil, PhD Universidad de Alcala, April 1997.
[5] N. Duffy, G. Lacey, Colour Profiling. Irish Machine Vision and Image Processing Conference (IMVIP-97), September 1997.
[6] N. Duffy, G. Lacey, Colour Profiling Using Multiple Colour Spaces. British Machine Vision Conference Proceedings (BMVC-98), September 1998.
[7] N.W. Campbell, B.T. Thomas, Automatic segmentation and classification of outdoor images using neural networks, International Journal of Neural Systems 8 (1) (1997) 137–144.
[8] J. Moreira, L. Da Fontoura Costa, Neural-based color image segmentation and classification using self-organizing maps, IX SIBGRAPI, Universidade Federal de Minas Gerais, Caxambu. Brasil, October 1996.
[9] T. Kohonen, Self Organizing Maps, Springer, Berlin, 1995.
[10] A. Gonzalez, M. Grana, A. D'Anjou, An analysis of the GLVQ algorithm, IEEE Transaction of Neural Networks 6 (4) (1995) 1012–1016.
[11] N. Karayiannis, J. Bezdek, N. Pal, R. Hathaway, P. Pai, Repairs of GLVQ: a new family of competitive learning schemes, IEEE Transactions on Neural Networks 7 (5) (1996) 1062–1071.