Design of Crack Detection System Software for IC Package Using Blob Analysis and Neural Network

Rosdiyana Samad[1], Mohd Rizal Arshad * [1], Zahurin Samad[2]
[1]School of Electrical and Electronic Engineering, Engineering Campus, Universiti Sains Malaysia, 14300 Nibong Tebal, Seberang Perai Selatan, Penang, Malaysia
[2]School of Mechanical Engineering, Engineering Campus, Universiti Sains Malaysia, 14300 Nibong Tebal, Seberang Perai Selatan, Penang, Malaysia

Abstract: In this research, three methods for the detection of crack defects on integrated circuit (IC) packages are proposed. These methods use blob analysis technique in image processing stage, and use multi-layered perceptron (MLP) neural network to classify the IC package. This paper presents the various filters and operations employed in blob analysis. The simulation results have shown, that a two-layer back-propagation neural network, which has a log-sigmoid transfer function in the hidden and output layer, could be trained to classify the IC package image. An early stopping technique was used in this study to provide benefits to the network performance in terms of a decrease in over-fitting. It was found that the optimal number of hidden neurons for the network 1,2 and 3 were 12, 12 and 10. The first method produced an accuracy of 74.82% with 87.72 ms processing time, while the second method utilizing the same classifier, achieved 86.17% accuracy with 119.45 ms processing time. The third method achieved 96.1% accuracy with 188.44 ms processing speed. The results obtained in this study indicated that the third method proceed better performance with higher accuracy and processing speed below 200 ms.

Key-words: IC package, Crack detection, Blob analysis, MLP neural network, Image processing

1. Introduction

In the semiconductor industry, machine vision is extensively used in various stages on the IC-device-manufacturing process [1]. The main automated inspection processes in IC manufacturing include mask and reticle inspection, in-process pattern inspection and final chip inspection for quality control [2]. Among the many types of IC package defect such as incomplete fill, void and off center; crack is the biggest problem faced by the manufacturer, especially the hairline crack. The hairline crack is difficult to be detected by human operator, and it can cause internal damage in the IC chip.

Crack usually occurs at the edge of the IC chip. Hairline crack has a different feature, which is a very tight breakage on package surface that may not extend through the package surface, and it is nearly invisible. The dominant cracking mechanism is moisture expansion due to thermal processing acting on concentrations of water vapor at the back of surface of the die-paddle, and at the front surface of the silicon die [3]. Fig 1(a) shows DIP (dual inline package) IC package with a hairline crack, while Fig 1(b) shows IC package with an apparent crack.

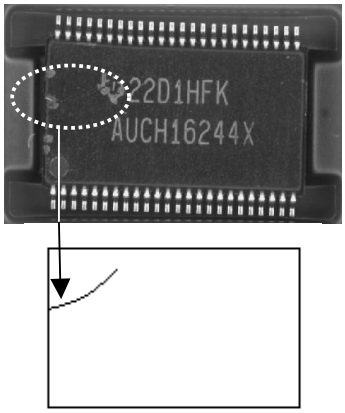

Fig 1(a). Hairline crack

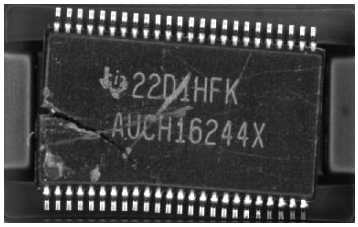

Fig 1(b). Apparent crack

The objective of this project is to design inspection system software that will automatically detect cracks on IC packages. The system uses the blob analysis technique in image processing stage, and followed by the back propagation neural network to classify the features. This paper presents three methods to detect crack on IC package, with each method adopts various approach in the image processing stage. Performance comparison was then conducted to determine the best method for the crack detection system. In this project, the image processing algorithm was developed using the LabVIEW™ software, while the network training was developed using MATLAB® software.

2. Approach and methods

All the three crack detection methods utilised a reference coordinate system, image masking, image processing and neural network technique. The procedure starts by the image extraction, where images stored in AVI format, are converted into individual image frames, in bitmap files. A total of 624 images consisting 260 defects and 364 non-defects images with 720x576 resolution pixels were used in the experiment.

2.1 A reference coordinate system.

Before the image-processing technique is applied, a reference coordinate system must be built in order to develop an automatic crack inspection system. A reference coordinate system is utilised to locate the reference feature in the inspection image, in order to move the region of interest (ROI) in relation to the object. The IC under inspection usually appears shifted or rotated within the image that need to be processed. This is because the size of the IC is smaller than the holding pocket. In this process, the ROI must be shifted and rotated the same way as the object.

In order to develop a reference coordinate system, we need to determine the *x* and *y* position, and orientation for coordinate reference of the object in the image using the edge detection method. This has been done by using LabVIEW™ function, i.e. *IMAQ Coordinate System (2 Rect)*. This function is used to compute a coordinate system based on the position of an object in a search area of an image [4]. Two rectangular search areas have been specified, each containing one separate, straight boundary of the object and the boundaries cannot be parallel, as shown in Fig 3.

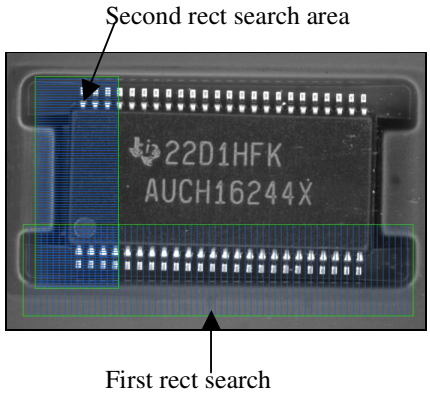

Fig 3. Rectangular search area

The first rectangular search area is the IC leads, which represented *x*-axis, while second search area is the vertical edge of IC package, which is represented *y*-axis. Then, the coordinate system axis direction is chosen. The outputs of this function are the origin, angle and axes direction of the coordinate system. To build a coordinate system for the first time, *Find Reference* mode in *IMAQ Coordinate System (2 Rect)* function is set. Next, the *Update Reference* mode is set to update the coordinate system in subsequent image. During this step, *IMAQ Coordinate System (2 rect)* function

locates the features in the search area and builds a new coordinate system based on the new location of the features. After a reference has been defined, the coordinate systems moves in relation to the object and the preprocessing automatically move the ROI to the correct position in the new coordinate system.

2.2 Removes IC leads, background, text & notch.

In the next step, the IC leads, notch, text images and background need to be removed, so that the detection process of crack become simpler. This was done by defining the regions to process with an image mask.

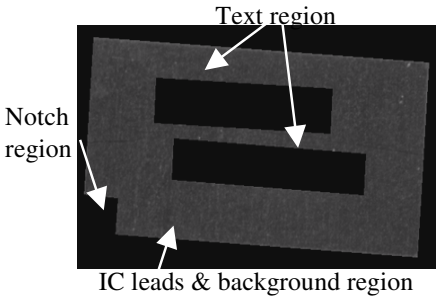

Fig 4. Removes IC leads, background, text & notch image.

In order to remove the IC leads and the background, the ROI was selected on the IC package, and the image was masked with the region outside the ROI. While, for the text image and notch, the ROI was selected on the text image and notch, with inter-ROI masking conducted. Fig 4 shows the masked image.

2.3 Image processing

Development of a reference coordinate system and image masking were conducted for all three methods. Blob analysis technique that contains a variety of operations in image processing stage to extract the crack features was applied across the methods. Blob (binary large object) analysis technique is used to extract the crack from the image, and to find the statistical information such as size of blob, area and the perimeter that were measured from the binary objects [4]. The image processing operations for all three methods are summarised in the Table 1. The extracted crack object is termed 'blob', after going through the image processing stage. Then, these blobs were measured to determine its features. All the measurements are summarised in Table 1. These measurements act as the input to the neural network classifier as shown in the Table 2.

2.4 Neural Network Classifier

624 data were utilised for the image processing stage, while another 332 data were used for the neural network classifier training. In order to make the data set for neural network, these data were restructured into matrices with 342 rows, and subsequently, imported to the MATLAB® software. The MATLAB® Neural Network Toolbox was used to construct the artificial neural network in this study.

In this study, the data matrix used for training the neural network was normalised to the range 0-1. In the training procedure, normalize data value was randomly inputted to the network. An early stopping technique was chosen in this study to acquire an acceptable generalization performance, which is used to prevent over fitting of the data by the network [5].

The collected data was divided into 4 subset. The first subset was the training set, which was used for computing the gradient and updating the network, weight and biases [6]. It has 206 data, which contains 96 defect image data, and 110 good image-data. The second subset was the validation set. The error on validation set was monitored during the training process. The third set was the test set. The test set error was not used during the training, but it was used to compare the different models. There were 68 data for validation and test set, which each set containing 36 good image-data and 32 defects. The forth set was the real data set, which is used to estimate

Table 1. Image processing techniques for method 1, 2 and 3.

Method	1	2	3
Image enhancement techniques	1. Brightness, contrast, gamma 2. Horizontal Kirsch filter 3. Low pass filter	1. Invert gray image 2. Laplacian filter 3. LUT: exponent 4. Gray morphology : proper close 5. Sobel filter	1. IMAQ Count Objects function 2. Find min pixel, extract & change value to 0 3. Convert new pixel value to image 4. Laplacian filter 5. Gray morphology : open 6. Gray morphology: erode 7. Median filter
Create binary image	Threshold (threshold values : 64-255)	Threshold (threshold values : 51-255)	Automatic threshold : Entropy
Improve a binary image	1. IMAQ Remove Particle function 2. Invert binary image	1. IMAQ Particle Filter function	1. Invert binary image 2. IMAQ Particle Filter function 3. Morphology : Dilate (2x7) 4. IMAQ Particle Filter function 5. Morphology : Dilate (1x5)
Make a particles/blobs measurement	**Parameters** : Blob area, number of holes, hole's area, hole's perimeter	**Parameter**: Blob area	**Parameters**: Coordinates *(min x, min y, max x, max y)*, width, chord & axes *(max intercept, mean intercept perpendicular)*

Table 2. Neural network input parameter for each method.

Number of method	1	2	3
Input parameter	1. Blob area 2. Number of holes 3. Holes' area 4. Holes perimeter 5. Blob area (from IMAQ function)	Blobs area at : 1. right side package 2. left side package 3. upper side package 4. lower side package	1. Blobs area 2. min x 3. max x 4. min y 5. max y 6. width 7. max intercept 8. mean intercept perpendicular

the network performance after training has finished. This set contains 182 good image data 100 defect image-data.

All the training simulations were performed with the same architecture. The architecture of the neural network chosen for this study was a multi-layer perceptron (MLP) model, with single hidden layer, and log-sigmoid transfer function in hidden and output layers. The training algorithm that was used in this study was Levenberg Marquadt algorithm. The algorithm is regarded as one of the fastest methods for training moderate-sized back propagation neural network [7]. It is a very efficient when training networks, which have up to a few hundred weights [8]. This algorithm has also shown high performance in function approximation problems.

For the network, the number of output neuron was 1, while the number of hidden neuron for each network is as shown in Table 3. The determination of the optimal number of hidden neurons was solved through trial and error approach. The network within 5 to 15 hidden neurons was simulated in order to determine the optimal number of hidden neurons. Although there seems to be no upper limit in the neural network using the early stopping algorithm, it was not necessary to use too many hidden neurons because it needs more computation time and memory requirement [6]. The network was trained for 30 iteration for each neuron, and then, the average of mean square error (MSE) was calculated. The neural network with the best performance was determined by minimum average of MSE. The values of minimum average of MSE are given in Table 3.

Table 3. The neural network simulation results for method 1,2 & 3

Network	Input neurons	Hidden neurons	Min average of MSE
Method 1	5	12	0.1803
Method 2	5	12	0.0701
Method 3	8	10	0.0565

3. Results and discussion

The most important factors in classifying crack inspection system are the classification accuracy and processing speed. Since the system must strive for real-time processing capabilities, the algorithms for all three methods need to be fast in execution. In this project, the processing speed was targeted for less than 200 ms, where effective methods of feature extraction technique must be adopted, to fulfill this requirement. Table 4 shows the results.

From the results, it can be seen that the first method produced the fastest processing time at 87.72 ms, with the lowest accuracy, at 74.82%. The second method achieved 86.17% accuracy with 119.45 ms processing time. The third method achieved the highest accuracy, with 96.1%, and the slowest processing time, at 188.44 ms. The results obtained in this study, indicated that method number three produced better performance with higher accuracy. Although the processing speed of this method is slower, it is below the 200 ms targeted processing speed.

Table 4. Speed processing time and classification accuracy results for each method.

Number of method	Speed processing time (ms)	Classification accuracy (%)
1	87.72	74.82
2	119.45	86.17
3	188.44	96.10

The third method shows that the apparent crack and scratch were accurately detected, where the smallest size for hairline crack that can be detected is within 0.05 mm to 0.075 mm. The first and second method can only detect hairline crack with size not less than 0.075 mm. Detection of hairline crack is a difficult task because of its intensity that nearly matches with intensity of IC's background.

All methods can convincingly detect apparent crack, and scratch defect. Although crack and scratch are two different kinds of defects, the manufacturer will reject scratch defect, as well as crack defect.

The execution time for each algorithm depends much on the used of central processing unit (CPU). In this study, all the algorithms were tested on CPU with Pentium 4, 2.66 GHz processor, and 256 MB random access memory (RAM).

4. Conclusion

This paper has presented the performance of three inspection methods, utilising MLP neural network classifier, for crack detection. The third method is found to be the best in detecting both kind of crack and also scratch defect. This method takes less than 200 ms to compute, showing that it is not computationally demanding and can easily be implemented into a real system. Nevertheless, admittedly, the capability of the system to search for crack defect is very much dependent on the image processing stage.

References:

[1] Kassim A A, Zhou H, Ranganath S, Automatic IC Orientation Checks. *Machine Vision and Application*, vol. 12, (2000), pp 107-112.

[2] Zhou H, KassimA A, Ranganath S, A Fast Algorithm For Detecting Die Extrusion Defects in IC Packages. *Machine Vision and Application*, vol.11, (1998),pp 37-41.

[3] Suhl, D., Thermally Induced IC Package Cracking. *IEEE Transaction on Component, Hybrids and Manufacturing Technology*, 13 (4), (1990) p. 940-945.

[4] IMAQ Vision Concept Manual National Instrument Corporation, (2000), USA.

[5] The MathWorks, Matlab and Simulink for Technical Computing. (2004) Available: http://www.mathworks.com/

[6] Prechelt L, Automatic Early Stopping Using Cross Validation: Quantifying The Criteria. *Neural Networks*, Vol. 11,(1998), pp. 761-767.

[7] Sohn J.H, Smith R, Yoong E, Leis J, Galvin G, Quantification of Odours From Piggery Effluent Ponds Using An Electronic Nose and Artificial Neural Network. Biosystems Engineering, Vol.4, no. 86, (2003), pp. 399-410.

[8] Hagan, M.T. & Menhaj, M. B., Training Feedfoward Networks with The Marquadt Algorithm. *IEEE Transactions On Neural Networks*, 5(6), (1994) pp. 989-991