Automated Inspection of PCB's using a Novel Approach

S. Burak Göktürk
Intelligent Systems Laboratory
Dept. Of Electric Electronic Eng.
Boğaziçi University

Lale Akarun
Dept. of Computer Eng.
Boğaziçi University

H. Işıl Bozma*
Intelligent Systems Laboratory
Dept. Of Electric Electronic Eng.
Boğaziçi University

Abstract

In this paper, we represent a novel approach to the automated inspection of printed circuit boards. A model based on the coordinates and connectivity analysis of the circles is formed using some new approaches to edge linking, and fusion of some edge based and region based algorithms.

A modified Canny edge detector is used as an edge based algorithm while an unsupervised learning algorithm is used to differentiate regions on the PCB. We have defined a membership function, which fuses the above two results.The edge linking algorithm extracts out the connectivity information for the circles using a new approach depending on making the decisions on fixation points.

Although the camera system works under imperfect illumination conditions, the performance has been satisfactory. The use of this new model and the novel techniques for region finding and edge following has proved to be efficient and time saving.

1. INTRODUCTION

Our problem is to detect defects on printed circuit boards. Typical defects that must be detected are over-etchings (opens), under-etchings (shorts), holes etc.Many different image processing techniques have been presented to solve the problem such as image subtraction [1],[8] and feature-based approaches [2]. Some approaches that use morphological operations have also been developed [3,4,5]. Other approaches are based on template matching [6]. Jarvis inspected PCB's using 5x5 binary templates [7]. Inspection methods that use design rules through graph matching have been developed [3] such as that[1] simulated by Darwish and Jain. In this system a semantic graph representing the circuit is obtained via morphological thinning followed by merging and deletion of some segments. In a system developed in [9], a segment graph representing the skeleton of circuit wires is compared with a library model in order to inspect wire width, seperation, and the desired circuit pattern.

This project presents a novel approach to the automated visual inspection of PCB's. The idea is to extract a minimum number of differentiating characteristics of a model PCB to detect the maximum number of errors on a PCB. Our objective is to test electrical connectedness. For this we concentrate on the locations of the circles and the paths that connect circles. By combining the information about the circles and the paths, we group the circles as nodes of a graph. The arcs between the circles represent the electrical connectivity.

The edge detector algorithm we use is a modified Canny edge detector. An unsupervised learning algorithm is used to differentiate regions on the PCB. We have implemented a membership function, that let the above two results be fused. The edge linking algorithm, using the fused data, extracts out the connectivity information for the circles using a novel approach depending on making the decisions on fixation points.

The paper outline is as follows: In section 2, we explain our approach in detail. In section 3, the hardware setup is described. In section 4, the experiments and their results are discussed. In the last section our conclusions are summarized.

2. Approach

Figure 1 shows the the flow of visual processing. The system works in two modes; 1) Learning mode in which a model PCB is inspected and the model for the PCB is created; and 2) Comparison mode in which a to-be-tested PCB is inspected and the model is formed. In both modes a preprocessing operation of logarithmic transform is applied after the image acquisition. Next, a modified Canny edge detector and an unsupervised grouping operations are applied in parallel and the results of the two operations are combined using a membership function. The graph of the PCB is constructed as the next step. If the system is in learning mode the golden graph model is recorded as the model library, if not a comparison of the graph with the model library is performed. All the steps will be explained in detail in the following sections.

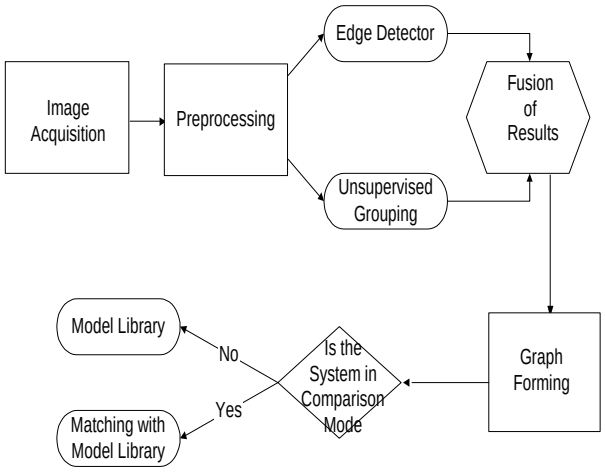

Figure 1 : Flow of Processing

* This research has been partially funded by TUBITAK-MISAG-65,1995, BU Research Fund Grant # 97HA0222, DPT 95K 120320.

2.1. Preprocessing

Due to nonuniform illumination conditions, it is impossible to differentiate between the four kind of objects on the PCB. So we have used a logarithmic transform [10] to compress the dynamic range. The applied transform is given as:

$$T(f_i) = \begin{cases} 4.6 + \log_2(f_i/24) & \text{if } 0 < f_i \le 24 \\ 74.8\log_2(f_i/24) & \text{if } 24 < f_i \le 255 \\ 0 & \text{otherwise} \end{cases}$$

f_i is the intensity of the ith pixel. The results of this transform is seen on Figure 2(b).

2.2. Unsupervised Clustering

Using an unsupervised learning algorithm, we aim to distinguish between the four different objects, that are circles, PCB background, paths, and the image background on a PCB. Due to inconstancy of illumination, a globally applied clustering algorithm fails to group different regions on a PCB. Instead,we have devised a block-based approach in which we apply clustering locally on 32x32 partially overlapping blocks. Within each block the algorithm that we use is the well-known K-means algorithm with K=4[15]. Let S be the clusters, C be the centroids of the clusters, d be a distance function, k be the iteration number. Then the algorithm we use may be summarized as follows :

1) $f_i \in S_i^k \quad iff \; d(f_i, C_i)^{k-1} < d(f_i, C_j)^{k-1} \quad for \forall j \; j \neq i$

2) $C_i^k = \frac{1}{N} \sum_{j \in S^k} f_j$

3) $if \; C_j^k \neq C_j^{k-1} \; \exists j, k = k+1 \; and \; goto \; 1.$

The selection of initial cluster centers is very critical for the success of the K-means algorithm. For the first block of the image, we choose uniformly spaced centers.The algorithm is iterated until convergence. The converged cluster center values are then used as initial values for the next block. This procedure is repeated left to right and right to left until all blocks are exhausted. Figure 2 (c) shows the result of unsupervised clustering on the image of figure 2 (b).

2.3. Edge Detection

The edge detection algorithm that we use is actually a modified Canny edge detector[7][10]. We process edge detection in 3 steps. First, the image is passed through a Gaussian low pass filter. Next, the derivative of the image is taken. Lastly, single thresholding is applied. Figure 2(d) shows the magnitude spectrum of the edges of figure 2(b).

Finally in order to track edges, angle info is extracted from the modified Canny edge detector. Figure 2(e) gives the angle information of edges. The angles are mapped to 256 intensity levels; with 0 degrees corresponding to 0 and the 2π degrees corresponding to 255.

2.4. Circle Detection

After unsupervised learning, we obtain four classes. First circle segments are identified based on the fact that the pixels belonging to the group with the highest mean intensity are those of circles. We then apply a connected

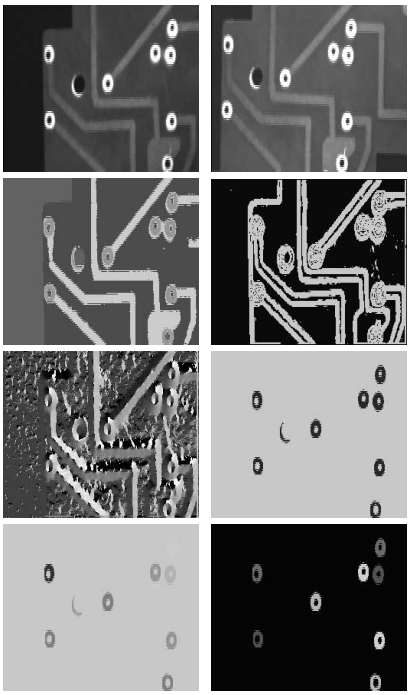

Figure 2 (a) Original image (b) After preprocessing (c) Result of Uunsupervised Clustering (d) Result of Edge Detector (e) Angle Information (f) Extracted Circles (g) Labelled Circles (h) Circles where noisy ones are eliminated

component analysis in order to differentiate the circles from each other. Figure 2(f) shows the result of the extraction of the circles from the unsupervised clustering operation of figure 2(c) and Figure 2(g) shows the result of the connected component analysis.

For the ith circle, the center coordinates $C_x(i)$ and $C_y(i)$ are found as:

$$C_x(i) = \sum_{n=1}^{N(i)} P_{nx} \qquad C_y(i) = \sum_{n=1}^{N(i)} P_{ny}$$

where N(i) is the number of all of the pixels in the ith circle and P is each pixel.

The x direction radius(r_x) and y direction radius(r_y) of each circle is found by the second moments of the pixels. This may be symbolized as follows:

$$r_x(i) = \sqrt{\frac{1}{N(i)} \sum_{n=1}^{N(i)} (P_{nx} - C_x(i))^2} \qquad r_y(i) = \sqrt{\frac{1}{N(i)} \sum_{n=1}^{N(i)} (P_{ny} - C_y(i))^2}$$

r_x and r_y are also the dimensions of an initial window on which a starting point on a circle is searched. In this analysis some noisy circles are found. In order to identify objects which are not really circles the following conditions are checked to obtain the circles of Figure 2(h) :

1- $r_x(i) > 3$ & $r_y(i) > 3$

2- $0.33 < r_x(i)/r_y(i) < 3$

2.5. Fusion Based Path Detection

The next step is to analyze the paths connecting the circles. For this, we have two sources of information : clustering and edge detection. Since neither is totally reliable by itself, we use a fusion algorithm to combine

the two data in order to determine the paths between the circles. We use a membership function to weight the results of unsupervised clustering of paths and to combine it with the results of the edge detection.

Unsupervised clustering algorithm assigns a point to one of four classes. A point is assigned to a class if the distance of the pixel to the corresponding class's centroid is less than the distance to any other centroids. There are cases in which this algorithm can not decide between two classes, or decides for one of the classes with a small margin. To extract information about the certainty of the algorithm, a membership function is developed which we call a membership certainty function. Similar approaches have been used in fuzzy systems using different membership functions.[13] If a point P is between C_k and C_{k+1}, the membership function M, is formed as follows:

$$M(P,C_k) = \frac{C_{k+1} - P}{C_{k+1} - C_k} \qquad M(P,C_{k+1}) = 1 - M(P,C_k)$$

The membership function for unsupervised clustering of figure 2(c) is given in figure 3(a). On this figure the bright pixels are the points where the clustering algorithm is more confident of its result.

A similar function is used to signify the confidence in path decisions. On figure 3(b) this function is implemented. The brighter the pixel, the greater is the probability of the pixel being a path point.

2.6. Edge Tracking

Once membership functions are computed, we look for a starting pixel for a path. The starting pixel is found by enlarging the initial window of the circle which is found by the unsupervised clustering algorithm. The starting point set is a two pixel set $X(x_1, x_2)$. In order to distinguish X, some considerations are made on x_1 and x_2. x_1 is a pixel that is on the window enlarged, and is assumed to be a circle pixel and x_2 is a point that is just outside the window, and is assumed to be a point on the path. x_2 should be a point with a membership function value of being a path point with a certain confidence such as a membership function value of 0.5 as in our tests, and also should go outwards from x_1. In order to ensure this outward direction, we check the angle of the edges at these points. There should be a difference of about $\pi/2$ in order for (x_1, x_2) to be a starting point set. Figure 3 (c) shows the starting points.

The next step after starting point set X is found, the contours on edges are constructed by tracking the edges through the direction of the angle. This algorithm links the edges through the direction of the angle of the edge checking whether that is a path point using the membership function. Figure 3 (d) shows the results of edge tracking.

In our edge linking algorithm, we have used a novel approach to make decisions about the edge's path. Through the direction of the edges, there are some points on which the angle information changes slightly with respect to the point tracked before it. These are actually critical points which we call fixation points. [14] Important decisions like arrival at a circle or a sharp turning of the edge, are made only on the fixation points. These points, which let us save considerable time, are

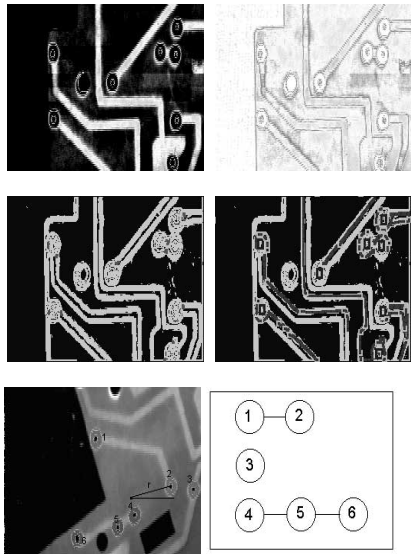

Figure 3(a)Membership function (b) Path Function (c)Starting Points(d) Tracked Edges (e) The PCB (f)The graph of the PCB

shown on figure 3 (d) as windows. When a fixation point coincides with a circle, we conclude that the circle is electrically connected to the circle on which the edge tracking has started. This information let us develop our algorithm based on a graph representation.

2.7. Model Formation

Our aim is to establish a model that is invariant to translations and rotations of the image. For this purpose, we use a graph based on a polar coordinate system. The center coordinates of each circle are modelled with respect to the average of the center coordinates of all of the circles. Figure 3(e) demonstrates our approach. The circles on the PCB are represented by a circle number, circle center coordinates in polar coordinates (r,θ) relative to the center coordinates of all of the circles, x directional radius(r_x) and the y directional radius(r_y). The circles connectivity is represented by a graph, each node of which is a circle. The whole graph consists of several connected subgraphs, and nodes $\{C_1, C_2,..,C_k\}$ signify the connected circles.

2.8. Graph Matching

This step is applied in the comparison mode. First, the circles of the library PCB and the to-be-tested PCB are matched one by one. Next, the graphs are matched.

In order to match the circles on the two PCBs, first, the number of circles on the library PCB and the tested PCB are compared. If the numbers match, the circles which are farthest from the centers of all circles in the library model and the tested model are found. An angle slip of rotation is calculated between these two circles. The angle slip is actually the core of the analysis in the sense that the rotation and the expected new coordinates of the circles on the to-be-tested PCB can be deduced. So all the other circles are checked through the angle slip found and if a match is found for each circle then the two PCBs

are matched. This gives us the transform T, that maps each circle in the to-be-tested PCB to the model PCB. That is to say :

$T(C_I) = C_k$ for each C_I in tested PCB and C_k in model PCB.

The procedure described above is repeated for each circle that is probable to be the farthest circle in the to-be-tested PCB until a match is found. Once T is obtained, it is applied to all of the circles in the graph of the tested PCB, so that the previously formed graph of the library PCB is the same with the newly obtained graph. If the two graphs completely match, this indicates a nondefective PCB.

3. Hardware

The visual processing is done in Intelligent Systems Laboratory, on the Smarteye Vision System which is designed around a high performance DSP chip TMS320C31PQL. To approach real time operation, the computationally intensive parts of the software will be converted to TI assembly code. The software is now written in C and cross compiled to TI assembly code. The illumination system consists of four lamps located so as to have uniform illumination. It must be noted that illumination is extremely important. In case of nonhomogeneous illumination, the system can give some faulty decisions in the unconstantly illuminated parts.

4. Experiments

Many experiments are applied on different PCBs. The different conditions of illumination normally do not effect the performance of the system in finding out the circles and matching them in the PCBs. Figure 4(a) and 4(b) which mutually correspond to the same PCB with different views, show the result of each circle matched. Each matched circle pair is shown with a different gray level.

As the board gets more complex, illumination causing deteriotion of performance becomes more crucial. This is primarily due to the erronous decisions made in the unsupervised clustering stage. These wrong decisions can lead to false edge trackings, but when suitable lighting is obtained, the graph matching is also successfully taken over. Figure 4(c) shows a faulty PCB with a broken path and Figure 4(d) shows how our approach eliminates this PCB through edge linking.

We have also tested our design with some synthetic images of unequally illuminated and noise added PCB images, one of which is given in figure 9(a).

In the tests two PCBs, that have been decreasingly illuminated in x,y or both directions have been used. Let the syntetic image S^x_{xy} be calculated from the original intensity image f_{xy} as follows:

$$S^x_{xy} = f_{xy} - f_{xy}(1-\rho)\frac{x}{w}$$

Where ρ is the illumination constant , and w is the width of the image.

Similar equations are applied to form illumination

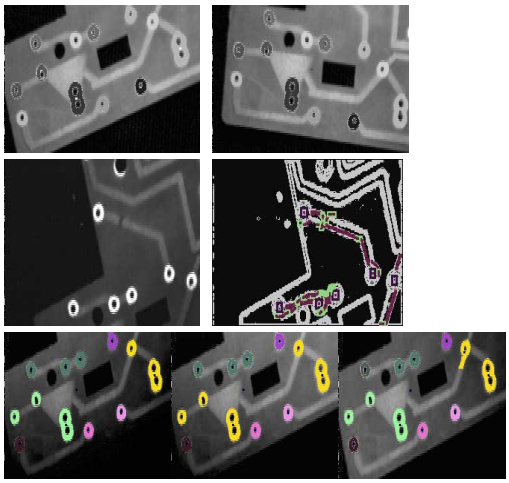

Figure 4 (a)The Model (b)To-be-tested PCB (c) A Faulty PCB (d) Elimination of the PCB (e) illumination with p=0.4 in x-dir (f) illumination with p=0.4 in y-dir (g) illumination with p=0.4 in x&y-dir

defects in y and both x and y directions. Three examples of synthetically generated images are given in figure 4(e),(f) and (g). The algorithm has been applied to two different synthetically illuminated PCBs containing 9 circles and 5 links in average. The results are given in Tables 1,2,and 3.Generally, the results of illumination decreasing in x direction is more satisfactory than the results of illumination decreasing in y-direction. This is the expected result due to our usage of unsupervised clustering with overlapping blocks travelling in x-direction.

Table –1 Results for different conditions of illumination changing in x-direction.

P	1.0	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2
% of unfound circles	0	0	0	0	0	0	0	0	5
% of missed radius	0	0	0	0	0	5	5	5	10
% of missed links	0	10	0	0	20	20	40	40	40

Table –2 Results for different conditions of illumination changing in y-direction.

P	1.0	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2
% of unfound circles	0	0	0	0	0	0	0	5	5
% of missed radius	0	0	0	0	0	0	0	0	5
% of missed links	0	10	10	10	10	20	20	30	40

Table –3 Results for different conditions of illumination changing in x & y direction.

P	1.0	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2
% of unfound circles	0	0	0	0	0	0	10	25	35
% of missed radius	0	0	0	0	0	0	5	0	0
% of missed links	0	0	0	0	0	30	50	60	70

Tests have also been applied to noisy images. Two different kinds of noise have been generated: Impulsive noise and gaussian noise. The noisy image n_{xy} is created from the intensity image f_{xy} as follows :

$$n_{xy} = \begin{cases} 255 & with \quad q \quad prob. \\ f_{xy} & with \quad (1-q) \quad prob. \end{cases}$$

Where q is the noise probability. Table 4 gives the results of tests on impulsive noise added images.

Table –4 Results for impulsive noise added images

q(%)	0.1	0.2	0.5	1.0	2.0
% of unfound circles	0	0	0	20	50
% of missed radius	0	0	0	0	0
% missed links	0	0	0	60	100

0-mean Gaussian noise is added to images with different standart deviations. The results are shown on table 5.

Table –5 Results for gaussian noise added images

Gaussian Parameter	0.2	0.5	1	1,5	2	3	5	7	10
% of unfound circles	0	0	10	0	0	0	0	0	10
% of missed radius	0	0	0	0	10	0	0	0	20
% of missed links	0	0	20	0	0	20	20	0	40

We have tested the system with different noise levels. For levels that are observable in an ordinary camera, such as (0.1%-0.2%), the system is uneffected by noise.

5. Conclusion

In this project, we present a new approach to the inspection of PCBs based on fusion of region and edge information. Our design consists of mainly four parts; unsupervised clustering, modified Canny edge detector, model forming based on data fusion and the model matching. Unsupervised clustering applied to the image by overlapping blocks, has proved to operate better than many other thresholding methods that we have tried. We have later fused the results of this unsupervised clustering with an edge-based segmentation algorithm. Our edge tracking algorithm has saved time by making tracking decisions only on fixation points. Our model for a PCB based on a rotation and translation invariant representation of circles and their connectivity checks the PCB's to be logically undefected.

We have developed our model on a real time system. On this system, we used a camera with a simple illumination set-up, which led to many problems due to nonuniform illumination. Illumination is actually the most important factor that diminishes the performance of our design. When good illumination conditions are satisfied, our design is robust.

References

[1] David T.Lee,A computerized automatic inspection system for complex printed thick film patterns, SPIE-Appl.Electron.Imaging Syst. 143,1978,172-177

[2] William K.Pratt, Image detection and registration, Digital Image Processing, pp.551-566;Wiley-Interscience, New York,1978.

[3] Ahmed M.Darwish and Anil K.Jain, A Rule Based Approach for Visual Pattern Inspection, IEEE Transactions on Pattern Analysis and Machine Intelligence,vol. 10,No 1,January 1988, pages 56-68.

[4] S.H.Oguz and L.Onural, An automated system for design-rule based visual inspection of printed circuit boards, in Proceedings of the 1991 IEEE International Conference on Robotics and Automation, Sacremento, CA, April 1991,pp.2696-2701.

[5] S.Birecik,M.Sezgin, I.O.Bucak, D.Demir,B.Sankur, E.Anarim, Baskili Devre Paketlerindeki Lehim Adacıklarının Matematiksel Morfoloji Yardımıyla İncelenmesi,Sinyal Işleme ve Uygulamalari 8-9 Nisan 1994

[6] A.Rosenfeld and A.C.Kak,Digital Picture Processing, Vol. I,Academic Press, Orlando, FL.

[7] J.Jarvis, A method of Automating the visual inspection of printed wiring boards,Image detection and registration, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. Pami-2,No.1,1980, pages 77-82.

[8] Y.Hara, H. Doi , K.Karasaki, T.lida, A system for PCB automated inspection using fluorescent light, IEEE Trans. Pattern Anal.Mach. Intell. 10(1), 1988, 69-78

[9] A.P.Sprague, M.J.Donahue, and S.J.Rokhlin, A method for automatic inspection of Printed Circuit Boards, CVGIP:Image Understanding 54(3),1991,401-415.

[10] Rafael C. Gonzalez, Richard E. Woods, Digital Image Processing, page 168.

[11] Robert M.Haralick,Linda G.Shapiro, Computer and Robot Vision Volume 1, page 305.

[12] John Canny, Computational Approach to Edge Detection, IEEE Transactions on Pattern Analysis and Machine Intelligence,vol. Pami-8,No. 6,November 1986, pages 679-697.

[13] J.C. Bezdek, Pattern Recognition with fuzzy objective function algorithms, Plenum Press, 1987.

[14] H. Yalçın, Işıl Bozma, An automated Inspection System with Biologically Inspired Vision, Proceedings of IROS98.

[15] Ramesh Jain, Rangachar Kasturi, Brian G.Schunck, Machine Vision, page 83.