Int J Adv Manuf Technol (2007) 35:293–300
DOI 10.1007/s00170-006-0730-0

ORIGINAL ARTICLE

Automated inspection of PCB components using a genetic algorithm template-matching approach

A. J. Crispin · V. Rankov

Received: 4 April 2006 / Accepted: 14 July 2006 / Published online: 25 October 2006

Abstract Automated inspection of surface mount PCB boards is a requirement to assure quality and to reduce manufacturing scrap costs and rework. This paper investigates methodologies for locating and identifying multiple objects in images used for surface mount device inspection. One of the main challenges for surface mount device inspection is component placement inspection. Component placement errors such as missing, misaligned or incorrectly rotated components are a major cause of defects and need to be detected before and after the solder reflow process. This paper focuses on automated object-recognition techniques for locating multiple objects using grey-model fitting for producing a generalised template for a set of components. The work uses the normalised cross correlation (NCC) template-matching approach and examines a method for constraining the search space to reduce computational calculations. The search for template positions has been performed exhaustively and by using a genetic algorithm. Experimental results using a typical PCB image are reported.

Keywords PCB manufacture · Component inspection · Template matching · Genetic algorithm

1 Introduction

Vision systems can be used to detect defects on surface mount device (SMD) printed circuit boards (PCBs) and many different inspection approaches have been developed (see for example [9, 12 and 8]). Surface mount techniques allow high component density boards to be manufactured at a high speed using automated equipment. Inspection is used to check for a range of possible errors that can occur during the manufacturing process.

The main manufacturing steps in the production of single-sided surface mount technology boards are shown in Fig. 1. The process steps involve applying solder paste to printed circuit board pads, placing surface mount devices on the board at correct positions and placing the board in an oven to solder components to pads (reflow soldering). Double-sided boards requiring the insertion of through-hole components undergo a more complex assembly process. Adhesive is used to hold the SMDs on the second side and wave soldering is used to solder through components.

Component placement can be achieved with pick-and-place automation where a vacuum nozzle is used for picking up SMD components from a tape feed and positioning these on the board. Vacuum blow off can lead to problems such as components not being placed (too low blow-off) and components being blown out of position (too high blow off) while poor X-Y mechanical alignment can result in position offset problems. Resulting component placement errors include:

(i) Missing components
(ii) Misaligned components
(iii) Rotated components

The solder reflow process can also cause components to be lifted and flipped due to unequal solder conditions and outgassing. Consequently, both inline and post inspection is used in the manufacture of SMD boards as shown in Fig. 1.

This paper explores automated object-recognition techniques for locating multiple objects (e.g. surface mount resistors) in an image to identify their position and rotation on the PCB board. This enables a component position

A. J. Crispin (✉) · V. Rankov
Faculty of Information and Technology,
Leeds Metropolitan University,
Headingley Campus,
Leeds, LS6 3QS, UK
e-mail: a.crispin@leedsmet.ac.uk

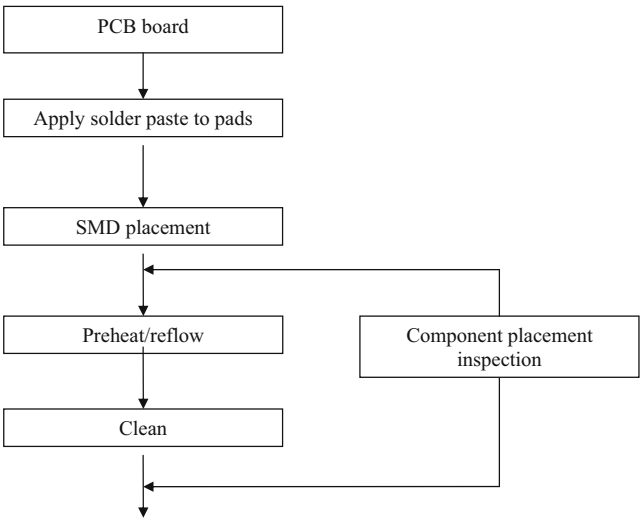

Fig. 1 Manufacture of SMD boards

quality check to be performed. The method developed is based on a template-matching and genetic algorithm search where a generalised grey-model template is used for multiple target recognition. Section 2 of the paper discusses the template-matching method and its use in object recognition. Section 3 discusses how a generalised template can be created for a set of components and generates the maximum likelihood search space to verify its use. Section 4 describes how the image search space can be constrained to reduce the number of calculations performed. In Section 5 a genetic algorithm solution for searching multiple objects in a source image is presented. Section 6 discusses the application of approaches developed and results obtained. Finally conclusions are drawn.

2 Template matching

Template matching is a method for identifying features in a source image that match a smaller sub-image called the template image [13]. It is commonly used in object-recognition applications. The basic template-matching algorithm involves sliding the template image over the source image and at each position calculating a grey-scale correlation measure using pixel intensities to estimate the degree of similarity between the template and source image region. Typically, the normalised cross correlation (NCC) is used in template-matching algorithms and is given by:

$$c(u,v) = \frac{\sum_{x,y}\left[f(x,y) - \overline{fu,v}\right]\left[t(x-u,y-v) - \bar{t}\right]}{\sqrt{\sum_{x,y}\left[f(x,y) - \overline{fu,v}\right]^2 \sum_{x,y}\left[t(x-u,y-v) - \bar{t}\right]^2}} \quad (1)$$

where

$f(x,y)$ is the matrix of grey-level pixel intensities in the source image

$\overline{fu,v}$ is the average grey-level intensity value of the source image in the region coincident with the template image

t is the matrix of grey-level pixel intensities in the template image

$\bar{t}$ is the average grey-level intensity value of the template image

The value of $c(u, v)$ ranges from −1 to 1 and is independent of scale changes of the source and template images. The maximum value of $c(u, v)$ indicates a position where the template best matches the source image. To extend normalised cross correlation to detect patterns that are rotated requires a new template-matching search for each angle, increasing the computational cost. The standard grey-level template-matching approach uses a single template to search for an individual component. The problem of locating and identifying similar components that exhibit variations of grey-level appearance requires a template-modelling approach [2].

3 Grey-model template

Figure 2 is a PCB source image showing a 7475 IC (a pin through hole (PTH) component) and six surface mount resistors. The quality requirement is to recognise and locate the six components located beneath the 7475 IC, which are either 5110 or 1001 surface mount resistors. Note that one of the 5110 resistors is rotated by 180° relative to the others. Each component has a fixed size and shape but has a different grey-level appearance due to differences in printing on the components and other differences related to markings of the component supplier etc. Non-uniform illumination can cause shadows that appear as shaded areas in the source image.

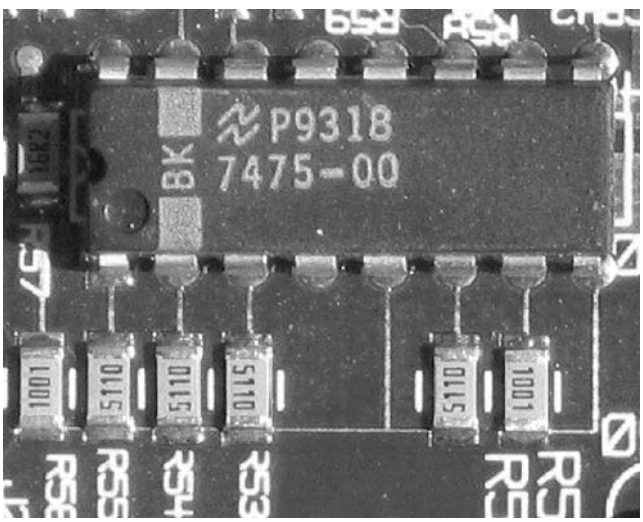

Fig. 2 PCB surface mount component image

The approach developed involves creating a grey-model template using a set of template images for each of the components to be located and recognised. This has been done by linearly combining a series of template images to average the statistical variation of grey-level intensity values that exists between each of the individual component images. The method extracts the same size of template images for each of the six components in the source image and computes the mean of corresponding pixel values in each of the template images. Figure 3 shows the generalised grey-model template image that has been created using six component template images of the same size extracted from Fig. 2.

The generalised template be can used to produce the maximum likelihood image of the search space as shown in Fig. 4. The maximum likelihood χ can be calculated using

$$\chi = [f - t]^T S^{-1} [f - t] \tag{2}$$

where S^{-1} is the inverse of the covariance matrix calculated using the matrix of pixel intensities in the source image region coincident with the template matrix and the matrix of template pixel intensities. In this equation, $[f-t]$ transposed is a 1×n matrix, and when multiplied by S^{-1} (which is an nxn matrix) produces a 1×n matrix. A scalar result is obtained when this 1×n matrix is multiplied with $[f-t]$ which is an nx1 matrix. The normalised value of χ ranges from 0 to 1. When χ tends to zero, there is a high likelihood that the template matches the corresponding source image region.

The maximum likelihood image has been produced by sliding the generalised template image shown in Fig. 3 over the source image (Fig. 2) and at each position calculating the covariance and subsequently the maximum likelihood using Eq. (2). The maximum likelihood search space image is obtained for a rotation angle of the template relative to the source image axis of zero degrees. Figure 4 shows a number of black regions where the maximum likelihood is near zero indicating a good likelihood of a match at positions in the image which correspond to components which are required to be located. Six dark regions corresponding to the surface mount resistor positions are shown indicating that the calculated generalised template is a good model for finding multiple objects in an image

Fig. 3 Generalised template

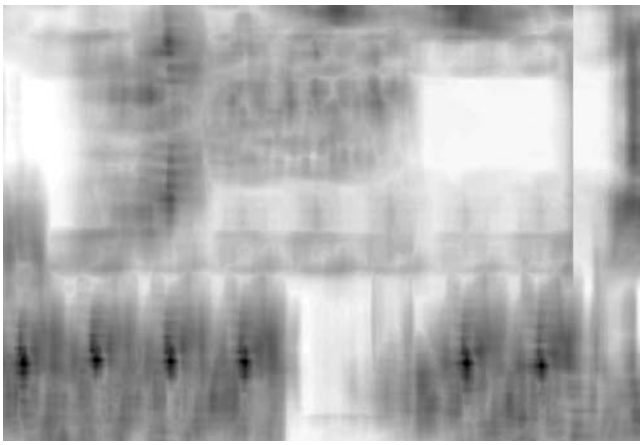

Fig. 4 Maximum likelihood image

4 Constraining the search space

A template search is required for locating and identifying multiple components in the source image. In the standard approach, the template would be located on every pixel in the source image to allow a similarity measure to be calculated using normalised cross correlation. Although normalised cross correlation is a good technique for detecting patterns in an image, it is based on summation and multiplication operations, which make it computationally time-consuming when searching a whole image. If ideal component centre positions on the PCB are known in advance (target points), then local searches can be performed around these positions by constraining the search to a local region centred on the target point. Using a small search area around the target points will lead to a faster search but will be more susceptible to errors (not finding objects) if the components are misplaced.

A misplaced component could be located at any position on the board (source image). If a pick-and-place removal machine is to be used to locate and remove it, then a search of the whole image would be required. Edge filtering of the source image can be used to constrain the search to edge positions and so reduce the search space. By only searching the image at edge locations reduces the number of NCC calculations that are needed to be performed. It is important that both strong and weak edges are detected so that templates can be accurately located. It was found that the Canny edge algorithm is successful at detecting weak edges compared to other edge filters such as Sobel [14]. The Canny filter was developed to improve upon other methods of edge detection [4]. The Canny method finds edges by first smoothing the image with a Gaussian filter, then performing a gradient calculation, thinning multi-pixel wide ridges down to a single pixel width (non-maximum suppression), applying low and high edge strength thresholds and finally accepting all edges over the low threshold

that are connected to edges over the high threshold. Figure 5 shows the source image that has been Canny edge filtered.

A vector of all edge positions can be extracted from the source image and plotted against the NCC coefficient calculated at each edge point using the generalised grey-model template. This is shown in Fig. 6 and represents the constrained search space. It has six peaks corresponding to the six resistors used to generate the grey-level template model. To use the Canny edge locations for locating templates requires coding the search algorithm such that the top left-hand corner of the template image is used as the reference point when performing the similarity calculation.

Edge detection also has the advantage that it is robust to changes in lighting conditions and contrast. Histogram equalisation can be used as a pre-processing step prior to edge detection to improve the dynamic range, contrast and as well as to maximise entropy (increase information) of the source image in situations where non-uniform illumination is used. To standardise images taken under different lighting conditions, histogram equalisation can be applied to the RGB components of the source image which are then subsequently recombined (see [6]). It is important to note that because our method is based on using a generalised grey-model template image that is produced by extracting same size sub-images, any histogram equalisation operation needs to be applied prior to creating the generalised template.

5 Search methods

The search for template matches can be performed exhaustively or by using a heuristic such as a genetic algorithm. A constrained exhaustive search performs a similarity calculation at every edge position found in the source image to find the best template matches. The criterion by which the search terminates is determined by the calculated NCC value. It has been found and validated empirically that an NCC value of 0.65 is a good termination criteria for stopping the search with a template match found.

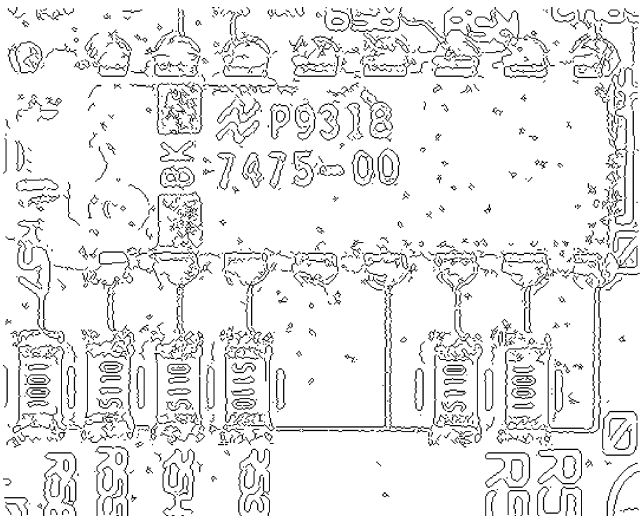

Fig. 5 Canny edges

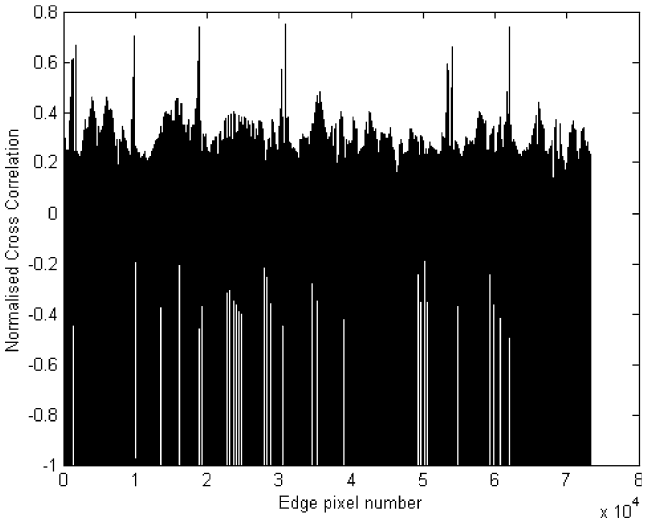

Fig. 6 Normalised cross correlation coefficient verse Canny edge pixel number

5.1 Genetic algorithm search

A genetic algorithm is a population-based probabilistic search algorithm. It uses operators modelled on the mechanics of natural selection and biological genetics [7, 10, and 11]. A genetic algorithm starts with the creation of a random population of solutions and a cost or fitness value (e.g. an NCC value) is calculated for each member. Using the fitness values as a guide, a subset of the population is selected as the parents. These parents are then combined to produce a new set of solutions called the offspring. This new set is used to replace a number of members of the original population and new cost values are calculated for the offspring. This process of choosing and using the fittest values to produce new populations is then repeated. As the population evolves, natural progression should yield better approximations to an optimal solution. A mutation operator is used to create new solutions by randomly altering a single solution string.

The genetic algorithm searches the constrained space (edge pixels) shown in Fig. 5. It is implemented using a population of indices-determining edge position (x, y) coordinates and a separate population of indices determining angle. The resistor objects can be present in the image at different angles of rotation. Consequently, an angle variable is defined that represents the angle of the template with respect to the source image axis. The resistors are usually placed at either 0, 90, 180 or 270° of rotation and so the angle vector is

restricted to these four values. The disadvantage of using just four angles is that the actual angle of rotation is sometimes not determined. The advantage of using four angles is that the computational overhead is kept low.

Each parent (i.e. edge position and angle indices) is assigned a fitness value by calculating the NCC value at the corresponding edge position (x, y) coordinates and angle. Roulette-wheel selection is used for choosing parents (index values), which are recombined using linear recombination to produce new offspring. Two parent solutions are linearly combined to produce two new offspring using the following equations.

$$\begin{aligned} offspring1 &= a(parent1) + (1-a)(parent2) \\ offspring2 &= (1-a)(parent1) + a(parent2) \end{aligned} \tag{3}$$

where a is a random weighting with values between 0 and 1. An extent value can be used to extend the range of the offspring value produced if required.

A genetic algorithm is good at targeting a single solution [3] and so a mechanism is required to allow the GA to find another target once a template match has been found. The approach developed involves changing the source image once a template match has been made. When a position in the source image is found where the NCC value is greater than 0.65 (a good template match is found), the region in the source image coincident with the template is replaced with an inverse template image so that future potential solutions in this region will generate poor correlation values. This approach effectively blocks out a region once a template match has been made. The algorithm terminates when all components have been found. This allows the search time performance to be directly compared to the exhaustive search.

The procedure for testing PCBs using genetic algorithms is presented in Fig. 7 and was implemented using Matlab. Firstly, the generalised template has been generated and the number of the objects to be found is set offline. Then, the image of the PCB under inspection is taken and the algorithm for finding objects and their positions and rotation angles is applied. If the positions and rotation angles are equal to their expected values (within noise limits), then the PCB under test passes the quality check.

6 Application and results

This section tests and compares the coding techniques developed for multiple object recognition using a generalised template. The results are shown in Table 1 and were obtained using Mobile AMD Athlon XP2500+ 1.06 GHz with 448 MB RAM.

An investigation was performed to compare the normalised cross correlation and the maximum likelihood template-matching methods using an exhaustive search of the unconstrained source image. The algorithm was tasked to locate the six 5110 and 1001 surface mount resistors in Fig. 2 using the generalised template shown in Fig. 3. The results show that the normalised cross correlation template-matching method is over four times faster than the maximum likelihood approach. With the maximum likelihood approach, minimum points are found sequentially, and once a minimum point is found it is assigned a 10×10 pixel area of white pixels to prevent multiple false solutions around that point. It was found that calculating the covariance matrix is very time-consuming and causes poor time performance compared to the NCC approach. A local search around each of the known component target points

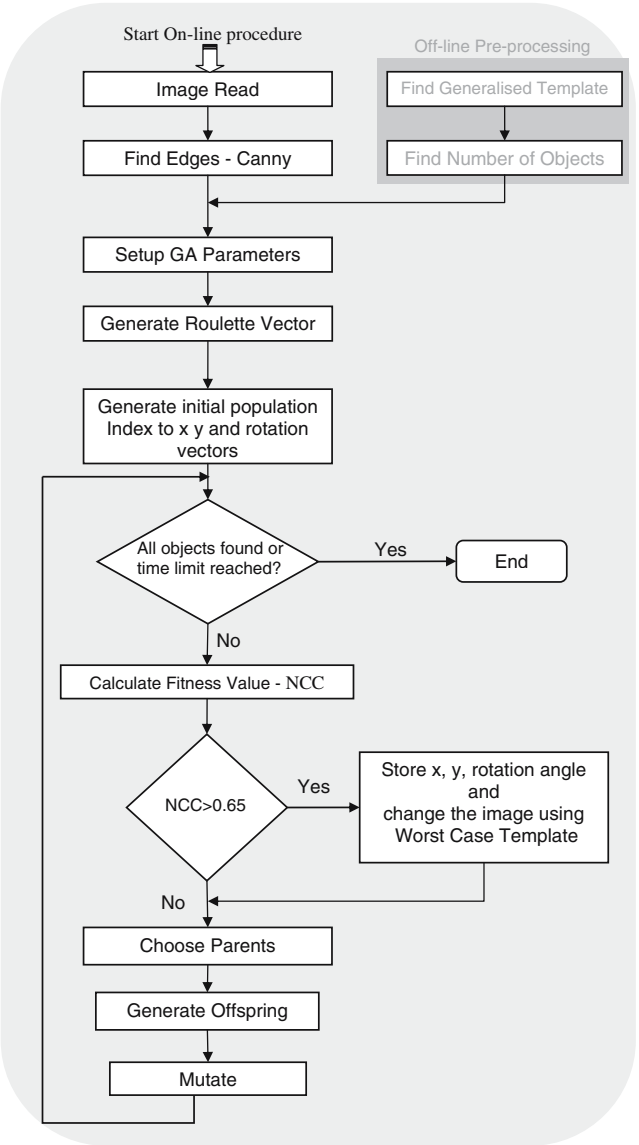

Fig. 7 Genetic algorithm template image search flowchart

was then performed within a predetermined rectangular region. A box containing ±25 pixels from the target position was used in the experimental trials as this provides a reasonable local search area for component misalignment. Restricting the search to local areas around known target points improves the search time considerably.

A local search around known target points will fail if the components are misplaced or blown out of position due to outgassing in the reflow process. In this case, a search covering the whole of the source image is needed to find the misplaced object. A previously discussed a Canny edge constrained search (Fig. 5) can be used to reduce the number of search points to be tested. Results were obtained for an exhaustive and genetic algorithm search of the Canny edges. Searching only the Canny edge point improves the exhaustive search time performance by more than a factor of three. It was found that the GA search for template positions is about six times faster than an exhaustive search of Canny edges. The GA search time is a mean of 100 runs performed on the same test data. The mean value of 39.5 s has a standard deviation of 33.1 s and was obtained using a population size that was set to 160, a recombination percentage of 75% and the mutation percentage to 35%. Genetic algorithms are non-deterministic algorithms and so the mean value is the important measure and relevant to a mass production situation where a large number of boards are being inspected. The GA mean value is better than values obtained for the other approaches.

In order to find an optimal parameter set for the genetic algorithm that minimises the execution time, an experimental statistical analysis has been performed. Initial trials were undertaken to determine the ranges of the controllable parameters, namely population size, recombination percentage, linear recombination extent and mutation percentage. Two sets of trial experiments were then performed for two consecutive ranges of values. The first set of experiments used population values 160, 200 and 240, recombination percentage values 75, 85 and 95%, mutation percentage values 15, 25 and 35% and linear recombination extents of 0, 0.1 and 0.2. The second set of experiments used population values of 80, 120 and 160; recombination percentage values 55, 65 and 75%, mutation percentage values of 25, 35 and 45% and linear recombination extents of 0, 0.1 and 0.2. In both experimental trials, the middle values represent nominal values and trials were carried out using full factorial design for the four parameters [1]. Each trial run was repeated ten times so that an experiment produced an output matrix of 17×10 values of the observed output parameter, namely algorithm execution time. Here, 17 is the number of runs with different parameters (16 runs for full factorial design of four parameters plus one for nominal values). Variations due to a random seed were blocked to ensure that changes in algorithm execution time are caused by parameter changes and not noise [5]. The best parameter set in each experiment was found by calculating the mean time and standard deviation for the ten runs in each of the experimental trials.

Table 1 Results

Method	Run time (s)
Maximum likelihood unconstrained exhaustive	3,303
NCC unconstrained exhaustive	747.2
NCC Canny edge constrained exhaustive search	230.8
NCC local search on component target points	47.5[a]
NCC Canny edge constrained GA search	39.5[b]

[a] Rectangle of ±25 pixels from the target position
[b] Mean of 100 runs with standard deviation of 33.1

The best parameters from the first set of experimental trials were found to be a population size of 160, a recombination percentage of 75%, a mutation percentage of 35% and a linear recombination extent of 0. The best parameters from the second set of experimental trials were found to be a population size of 160, a recombination percentage of 75%, a mutation percentage of 45% and a linear recombination extent of 0. The same best values for population size, recombination percentage and recombination extent were obtained from both trials. The only ambiguity was the value to use for mutation percentage and further trials were done to decide on its value. These revealed that a mutation rate of 35% gives the best algorithm execution time with all other parameters set to their optimal values. In summary, the best parameter set was found to be a population size of 160, recombination percentage of 75% and mutation percentage of 35% and these values were used in the comparative tests shown in Table 1.

Figure 8 plots, in a typical run, the difference between the total number of objects to be located and the current number found against iterations. In Fig. 8, N is defined as:

$$N = \frac{N_{TOTAL} - N_{FOUND}}{N_{TOTAL}} \tag{4}$$

where
N_{TOTAL}=Total number of objects to be located
N_{FOUND}=Current number of objects found

Figure 8 shows a series of steps reducing to a value of zero when the total number of objects to be located equals the number of objects found. A new object (surface mount resistor) is detected on each step. The genetic algorithm terminates when all required objects have been found and returns the (x, y) positions found for each component together with its angle of rotation, which is restricted to four angles (0, 90, 180 and 270°). A quality check can be

made using the values returned by the algorithm. The algorithm will not generate a correlation value greater than 0.65 for a component that is rotated with respect to the four angle positions allowed and so will not be recognised. Consequently, a component rotated out of position will not be recognised so that the total number of objects detected can be used as a quality check. Figure 9 shows the source image with all six components located using the genetic algorithm multiple object recognition method.

The key factor regarding the scalability of the approach is the construction of the generalised template. In this work, the generalised template is constructed from six component templates of the same size extracted from the source image of the same object with variations in labelling characteristics. Consequently, the approach is limited to searching up to six objects. Figure 10 plots the mean search time against the number of objects being searched in the image for the NCC Canny edge GA method. Results for both independent and simultaneous searches are shown. A simultaneous search is one which searches for all objects at the same time, while the independent approach searches for one object at a time. The results show that the GA simultaneous search approach produces better mean time values as the object number increases and, consequently, is a better approach for multi-object recognition.

The NCC local area search around component target points works reasonably well but assumes that target points are known in advance and that the source image is aligned to these target points. The advantage of the GA approach is that because it is a global search on edges it is not dependent on camera alignment. This is an important point as in a production environment it is not always possible to guarantee camera alignment. In both cases, if six resistors are located, the PCB passes the QA inspection criteria.

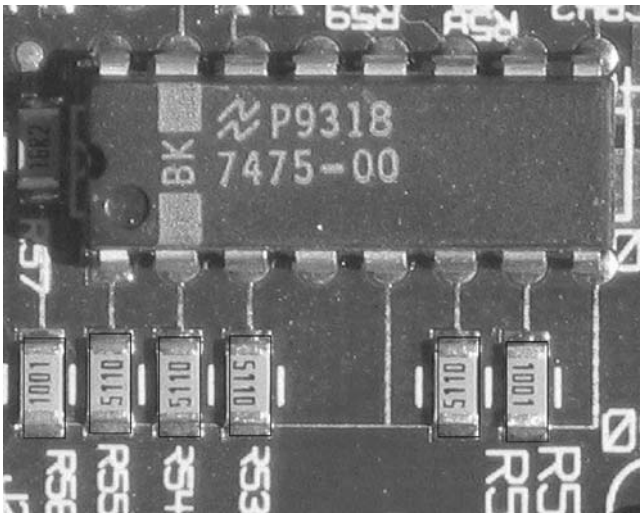

Fig. 9 Source image with six components recognised and located using a genetic algorithm

7 Conclusions

This research developed methods for multiple-object recognition and applied these to the problem of finding the positions and angles of surface mount resistors located on a PCB board to enable a quality control check to be performed. The basic approach is based on creating a generalised grey-model template for a set of components. The search can be constrained to local regions around known target points for the components or, if the whole source image is to be searched for a misplaced component, then the search can be constrained to Canny edge positions.

If the component target points are known in advance, then it is practical to exhaustively search sequentially small areas around each of the target points to find a template match. If the whole image is to be searched, then

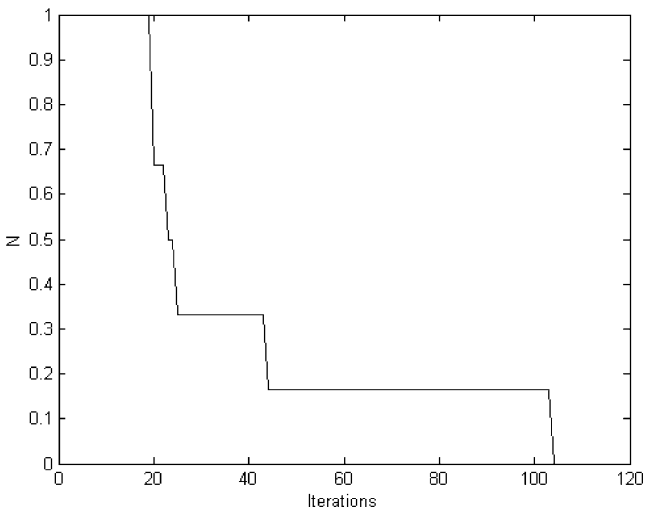

Fig. 8 Objects located versus iterations

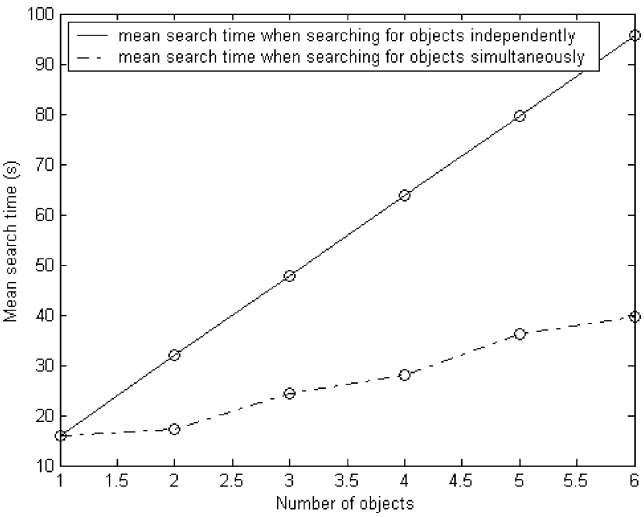

Fig. 10 Scalability results for the genetic algorithm approach

constraining the search space to Canny edges improves the exhaustive search time performance by a factor of three compared to searching the raw image.

To enable the genetic algorithm to recognise multiple targets, a method was devised to refocus the population search once a template match has been made. This involves replacing regions in the source image coincident with a successful template match with an inverse template image to prevent finding another solution in that region. The genetic algorithm terminates when all requested components have been found (in every test the GA finds all components) and has a search time performance that is about six times faster than an exhaustive search of the same constrained search space and is faster than a series of local exhaustive searches around known target points. Because the genetic algorithm approach searches on edges in the whole image, it does not depend on image alignment (which is necessary when comparing two images for defects). The methods developed in this work have many other potential applications including robotic vision systems and satellite surveillance.

References

1. Box G, Draper N (1987) Empirical model-building and response surfaces. Wiley, New York
2. Cootes TF, Page GJ, Jackson CB, Taylor CJ (1995) Statistical grey-level models for object location and identification. Proc British Machine Vision Conf. BMVA Press, pp 533–545
3. Crispin AJ, Rankov V (2002) Part recognition using genetic algorithms. 18th National Conference on Manufacturing Research, pp 273–278
4. Canny J (1986) A computational approach to edge detection. IEEE Trans Pattern Anal Mach Intell PAMI-8(no 6):679–698
5. Czarn A, MacNish, Vijayan K, Turlach B, Gupta R (2004) Statistical exploratory analysis of genetic algorithms. IEEE Trans Evol Comput 8(4):405–421
6. Finlayson G, Hordley S, Schaefer G, Tian GY (2005) Illumination and device invariant colour using histogram equalisation. Pattern Recogn 38(2):179–190
7. Goldberg DE (1989) Genetic algorithms in search optimisation and machine learning. Addison Wesley Longman Inc, Boston
8. Hata S (1990) Vision systems for PCB manufacturing in Japan. IECON '90, 16th Annual Conference of IEEE Industrial Electronics Society, pp 792–797
9. Horng-Hai Loh, Ming-Sing Lu (1999) Printed circuit board inspection using image analysis. IEEE Trans Ind Appl 35(2): 426–432
10. Mitchell MA (1996) Introduction to genetic algorithms. MIT Press, Cambridge
11. Michalewicz Z (1999) Genetic algorithms + data structures = evolution programs. Springer, Berlin Heidelberg New York
12. Moganti M, Ercal F, Dagli CH, Tsunekawa S (1996) Automated PCB inspection algorithms: a survey. Comput Vis Image Underst 63(No 2):287–313
13. Seul M, O'Gorman L, Sammon MJ (2000) Practical algorithms for image analysis: description, examples and code. Cambridge University Press, Cambridge
14. Sonka M, Hlavac V, Boyle R (1999) Image processing, analysis, and machine vision, 2nd edn. PWS Publishing, Boston