Proceedings of the 2003 IEEE
International Conference on Robotics & Automation
Taipei, Taiwan, September 14-19, 2003

Locating and Checking of a BGA Pin's Position Using Gray Level

Chi-Wei Ruo Ching-Long Shih

Department of Electrical Engineering
National Taiwan University of Science and Technology
43, Section 4, Keelung Road, Taipei, Taiwan, 106, R.O.C.
Email: shihcl@mail.ntust.edu.tw

Abstract

A machine vision system for SMT-mounting machine applications is usually a two-stage algorithm. It first measures the centroid and rotation angle of the SMD, and then checks each pin's area, position error, and grid coordinate location. In this paper, a set of complete procedures is proposed to locate and check the BGA image. During the locating procedures, we first calculate a threshold of the frame using an iterative threshold algorithm. If an object is found under this threshold, then the pin's area is calculated by the local threshold by using a momentum algorithm. After that, whether this object is a pin or not is decided by its neighbors' relative positions, the approximate rotation angle for finding the outer pins is calculated, and the centroid as well as the rotation angle of a BGA component is calculated by the rectangular least squares algorithm. The checking procedure also measures each pin's area using the momentum algorithm, then it calculates the radius of the moving sum using each pin's area, and finally measures the position error using a moving sum algorithm and judges each pin's type by gray level. The new method uses the gray level statistic information to solve the empty pad's problem and utilizes the symmetrical property of a circle to deal with the shape problem. Lastly, the CRC algorithm is used to check the correspondence between each pin and its pin type. This new method has a high accuracy and a low execution time. It can meet the crucial timing requirement of a high-speed SMT machine through experimental verification.

1. Introduction

The main function of an SMT (Surface Mounting Technology) mounting machine is to place the SMD (Surface Mounting Device) on the PCB through picking up and vision locating. Machine vision is used to measure the centroid and rotation angle on a variety of components. A machine vision program for an SMT-mounting machine application is a two-stage algorithm. It measures the centroid and rotation angle first, and then checks each pin's area, position error, and grid coordinate. The first stage must be robust enough to get the data when the picture is stained or corrupted, and the second stage must have both good accuracy and low execution time [1, 2].

There are already many research studies that focus on PCB checking or BGA PCB checking [3-7], but research that focuses on a mounting machine's vision problems are few. Barrtman [8] used a lead points' gray levels to calculate an SMD's centroid. Burel [9] applied neural networks to estimate an SMD's centroid. Hata used the variation of feature vectors to establish multiple dimension vectors and applied its difference to recognize an IC [10].

In this paper we propose a new algorithm for a high-speed mounting machine's BGA component locating and checking procedures. During the locating procedures, we first calculate a threshold of the frame using iterative threshold algorithm [14]. If an object is found under this threshold, then we calculate the pin's area by local threshold using a momentum algorithm [13]. After that, we decide whether this object is a pin or not by its neighbors' relative positions, and then we can calculate the approximate rotation angle for founding the outer pins, the centroid, and rotation angle of a BGA component which can be calculated by a least squares algorithm. When all pins are found, we use the new proposed least squares rectangle algorithm to calculate the BGA's centroid and rotation angle, which has a faster calculation speed and more robustness than using the 4 individual least squares lines method [3].

During the checking procedures, an iterative threshold is used to find some suspicious objects among a frame. If the distance from an object to its neighbors is one pitch and the lines that go through this object and its neighbors are perpendicular, then we can make sure that this object is a BGA pin. We propose a new method to measures the BGA pin's coordinate by a moving sum algorithm in the gray frame. The pin's type is also examined using its gray level statistic information.

All the binary algorithms in the locating and checking procedures are only used to judge the existence of a pin and to remove some fault objects, so that binary algorithms will not participate in the calculations which concern the numerical accuracy. Moreover, we put the geometrical information "BGA pin looks like a circle" into the algorithm to eliminate some noise from the BGA pin's image. Finally, each BGA pin type is verified by CRC algorithm[17], and each pin's position error is calculated. This proposed new BGA image locating and checking method has a high accuracy and a low execution time.

2. Measuring the center and rotation angle

The BGA locating algorithm first uses an iterative and momentum threshold algorithm to remove some fault noise, and then uses radial research to reduce the search time, and finally uses a least squares rectangle algorithm to measure the center and rotation angle of the BGA. We

first present the key algorithm for obtaining the center and rotation angle of a BGA chip.

2.1 Least squares rectangle algorithm

We use outer rectangle pins here to calculate the position and then implement the least squares rectangle algorithm. This method first finds the pins on the BGA's outer rectangle and then classifies them into the groups U, D, R, and L. A rectangle can be described as four equations

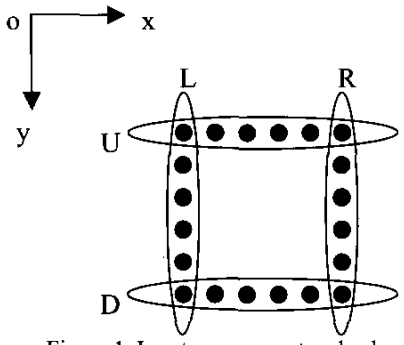


Figure 1. Least squares rectangle algorithm.

$$x + my + C_L = 0 \tag{1}$$

$$x + my + C_R = 0 \tag{2}$$

$$mx - y + C_u = 0 \tag{3}$$

$$mx - y + C_D = 0\,. \tag{4}$$

then

$$m = \left(\frac{\sum_{p\in U} xy + \sum_{p\in D} xy - \sum_{p\in R} xy - \sum_{p\in L} xy - \frac{\sum_{p\in U} x \sum_{p\in U} y}{n_U} - \frac{\sum_{p\in D} x \sum_{p\in D} y}{n_D} + \frac{\sum_{p\in R} x \sum_{p\in R} y}{n_R} + \frac{\sum_{p\in L} x \sum_{p\in L} y}{n_L}}{\sum_{p\in U} x^2 + \sum_{p\in D} x^2 + \sum_{p\in R} y^2 + \sum_{p\in L} y^2 - \frac{\left(\sum_{p\in U} x\right)^2}{n_U} - \frac{\left(\sum_{p\in D} x\right)^2}{n_D} - \frac{\left(\sum_{p\in R} y\right)^2}{n_R} - \frac{\left(\sum_{p\in L} y\right)^2}{n_L}} \right) \tag{5}$$

$$C_U = \frac{\sum_{P\in U} (y - mx)}{n_U} \tag{6}$$

$$C_D = \frac{\sum_{P\in D} (y - mx)}{n_D} \tag{7}$$

$$C_R = \frac{\sum_{P\in R} (-my - x)}{n_R} \tag{8}$$

$$C_L = \frac{\sum_{P\in L} (-my - x)}{n_L}, \tag{9}$$

where terms n_U ,n_D ,n_R, and n_L are the number of pins for each group U, D, R, and L, respectively. The centroid of the BGA is then located at

$$x_o = \frac{C_R + C_L + m(C_u + C_D)}{-2 - 2m^2} \tag{10}$$

$$y_o = \frac{-C_u - C_D + m(C_R + C_L)}{-2 - 2m^2}\,, \tag{11}$$

with a rotation angle of

$$\theta = \tan^{-1} m\,. \tag{12}$$

2.2. BGA locating flowchart

The program flowchart for BGA locating is shown as Fig. 2. First, we calculate a threshold of the frame using an iterative threshold algorithm [14]. Because the gray level's distribution range of a BGA pin is wider than the background's, the variance in the gray level of the BGA pins is bigger than the background gray level's variance. If the background is dark and the object is bright, then when we binarize this frame using the iterative threshold algorithm, the BGA pins' size will shrink. We find the outer bound by a radial search using this threshold.

If an object is found, then we calculate the pin's area by another local threshold using momentum algorithm [13] with the ROI being 1 pitch by 1 pitch. After that, we decide whether this object is a pin or not by the lengths and relative angles between its neighbors. Since a tin ball's diameter may change when the illumination changes, it is not reliable to judge an object by only using its area's size. A BGA pin must have at least 2 neighbors within its 4 adjacent neighbors and its neighbors must lie on the intersection of the grid lines.

We judge whether this object is a pin or not by the relative angles between 2 neighbors. When all pins are found, we calculate the approximated BGA rotation angle according to its neighbor's relative position. We then can get its outer rectangle pins using the approximate rotation angle. Finally, we calculate the rotation angle and position offset using the least squares rectangle method.

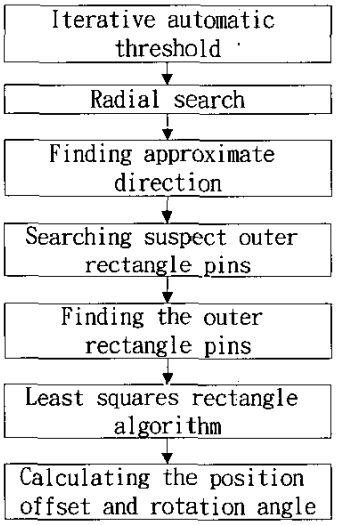


Figure 2. The program flowchart for BGA locating.

3. BGA image checking

In BGA image checking, the items checked for each pin are its area, center position error, and pin type on a grid coordinate position. The acceptable error distance square in this system is below 1 $Pixel^2$. The checking procedures first measure each pin's area using the momentum algorithm and labeling algorithm [16], and then decide whether this image is a null background, empty pad, or normal BGA pin by its gray level and area size (see Fig. 3). It calculates the radius of the moving sum under each pin's area and measures the pin's position error by the proposed sub-pixel moving sum algorithm (which will be discussed later). Finally it uses the CRC (cyclic redundancy check) algorithm [17] to check the correspondence between each pin and its pin type.

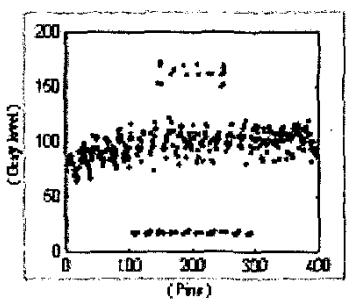


Figure 3. BGA pin's gray level: the brighter pins are empty pads, the lower parts are null but only background, and the normal pins' gray levels are between them.

3.1 Sub-pixel moving sum algorithm

Because a BGA pin looks like a circle, if a line passes through the circle, then we can calculate the moving sum along this line with the summation range of one BGA pin's diameter. The perpendicular line that goes through the midpoint of the maximum moving sum's interval must also go through the circle's origin. If we utilize this property to find out two independent perpendiculars, then the position of the BGA pin can be found at the intersection point.

The moving sum method achieves only pixel accuracy, but this accuracy is not good enough to satisfy the requirement and its accuracy must be improved to a sub-pixel's extent. The method to do that is using linear interpolation to get the gray levels between the two adjacent pixels and then calculate the maximum value of the sub-pixel moving sum. The moving sum $S(k)$ of n pixels along a line is defined as

$$S(k) = \sum_{i=k}^{n+k-1} G_i \cdot \tag{13}$$

As shown in Fig. 5, the solid circles represent the gray levels of the pixels, and the horizontal line's width represents a pixel's width. If the moving sum along a line has two or more maximums in different locations, then we choose the greatest k as its output. When the maximum of a moving sum is found at position x, then we can obtain

$$S(x) > S(x+1) \quad \text{and} \quad S(x) \geq S(x-1);$$

and hence,

$$G_{x+n-1} > G_{x-1} \quad \text{and} \quad G_x \geq G_{x+n}.$$

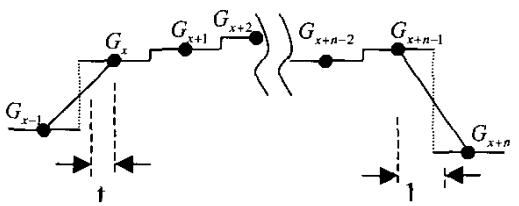


Figure 4. The moving sum along a line.

Because x is an integer, the fraction accuracy cannot be achieved. Therefore, we add t (where $0 \leq t < 1$) pixels' gray levels at the front of the moving sum and (1-t) pixels' gray levels in the rear using a linear interpolation method as shown in Fig. 4, the fraction's moving sum $S_x(t)$ can be obtained as

$$S_x(t) = tG_x - \frac{t^2}{2}(G_x - G_{x-1}) + (1-t)G_{x+n-1}$$

$$+ \frac{(1-t)^2}{2}(G_{x+n} - G_{x+n-1}) + S(x) - \frac{G_x + G_{n+x-1}}{2} \tag{14}$$

Because x is a constant, setting $S_x'(t) = 0$ yields

$$t_{\max} = \frac{G_x - G_{x+n}}{G_x - G_{x-1} + G_{x+n-1} - G_{x+n}} \tag{15}$$

and

$$0 < t_{\max} \leq 1.$$

Therefore, the midpoint x_c of the maximum fraction moving sum's interval is

$$x_c = x + \frac{n}{2} - t_{\max}. \tag{16}$$

As a result, we have the following theorem.

Theorem 1

If n points integral moving sum $S(x)$ has a maximum, then n points fraction moving sum $S_x(t)$ must have a maximum at $t_{\max}$, and $0 < t_{\max} \leq 1$.

3.2 BGA pin's center position measurement

A single BGA pin's image is rather small and is often strained by noises. It is therefore necessary to calculate the moving sum from different directions to reduce the noise's influence. Choosing the 4 moving sum directions (0° , 90° , −45° , and 45°), and the directions of the perpendiculars are (0°, 90°, −45°, and 45°). Although 4 lines may generate 6 intersected points at most, we choose the point with the shortest distance

among 3 lines as the output location. Selecting arbitrarily 3 lines among these perpendiculars, then an isosceles perpendicular triangle is generated. The shortest distance among the 3 lines is proportional to the triangle's shorter side length (Fig. 5).

Selecting 3 lines among 4 lines produces 4 situations. We calculate the short side length of the triangles for each situation and choose the shortest one, and then the circle origin can be found as shown in Table 1. The distances from this point to the 3 perpendiculars are all the same, but the distance to the fourth perpendicular is longer, and so the fourth perpendicular is treated as noise. A moving sum is sensitive to its accumulation's radius and each pin has a different area. Thus, we use its area to calculate its radius.

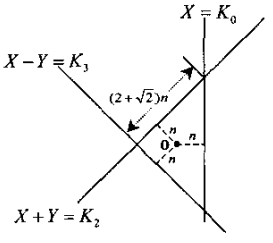


Figure 5. Isosceles perpendicular triangle, where the 4 perpendiculars at the midpoints be $X = K_0$, $Y = K_1$, $X + Y = K_2$, and $X - Y = K_3$.

Table 1. The shortest side length and circle origin

Line removed	Triangles' shortest side length	Circle origin (X, Y)
0°	$\frac{\lvert 2K_0 - K_2 - K_3 \rvert}{\sqrt{2}}$	$\left(\frac{\sqrt{2}-1}{2}(K_2+K_3)+(2-\sqrt{2})K_0, \frac{K_2-K_3}{2}\right)$
90°	$\frac{\lvert 2K_1 - K_2 + K_3 \rvert}{\sqrt{2}}$	$\left(\frac{K_2+K_3}{2}, \frac{\sqrt{2}-1}{2}(K_2-K_3)+(2-\sqrt{2})K_1\right)$
45°	$\lvert K_0 - K_1 - K_3 \rvert$	$\left(K_0+\frac{-K_0+K_1+K_3}{2\sqrt{2}}, K_1+\frac{K_0-K_1-K_3}{2\sqrt{2}}\right)$
−45°	$\lvert K_0 + K_1 - K_2 \rvert$	$\left(K_0+\frac{-K_0-K_1+K_2}{2\sqrt{2}}, K_1+\frac{-K_0-K_1+K_2}{2\sqrt{2}}\right)$

3.3 BGA pin's arrangement check

Inside a BGA component, the pins are placed in a two-dimensional grid array, and it is necessary to check whether each pin is placed in the right grid coordinate. When we check a BGA pin using machine vision, the result is among one of 3 types: null, empty pad, and normal BGA pin. It is quite intrusive to save every pin's grid coordinate and type when checking, and this method causes two problems. First, it wastes memory space. Secondly, different components take different memory sizes. A database with a fixed size for each record is more reliable and faster, and the complexity of programming will become much easier.

The arrangement checking can be treated as an error-detecting problem. The most general algorithms for solving the error-detecting problems are parity check, checksum, and CRC. Among them, CRC has the highest accuracy. A 32-bit CRC is implemented here, and the fault acceptance rate is about $1/2^{32}$. Because the CRC's error-checking bytes have a fixed length, the programmer can use a database with a fixed size for each record. This method increases speed, reduces complexity, and the system's performance improves.

3.4 BGA image checking algorithm

The flowchart of checking a BGA image is shown in Fig. 6. The checking procedure first gets a single pin's image with the size of one pitch square, measures its area by a labeling algorithm, then decides whether this image is a null background, empty pad, or normal BGA pin by its gray level and area size, uses the moving sum algorithm to measure its center position, and then checks the pin's type on the pin grid coordinate position. After the calculation of each grid line's equation by the least squares error, we can obtain the intersection of these grid lines. We then calculate the pin's position error as the distance between measured pin's center position and one of the nearest grid line intersection (see Fig. 7). Finally, we decide whether it is a qualified component or not.

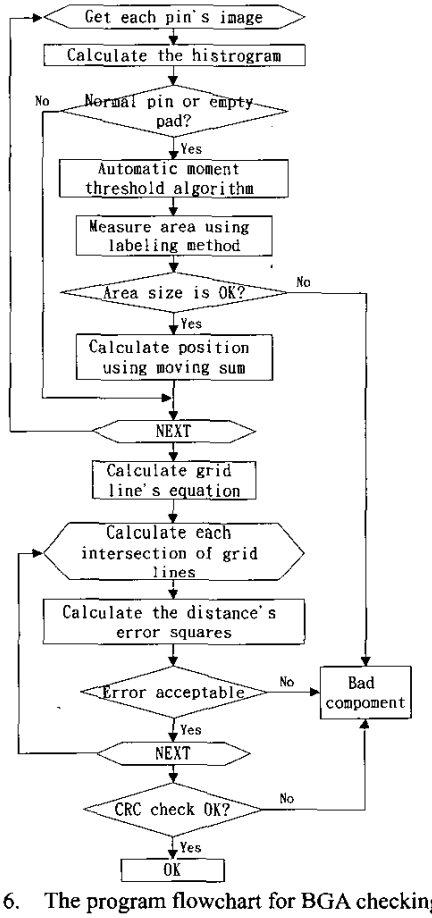


Figure 6. The program flowchart for BGA checking.

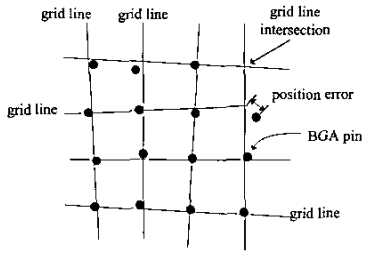


Figure 7. BGA pin position error.

4. Experimental results

The experimental image system's image capturing card is a Leutron PIC-PORT MONO H4 and the camera is a SONY XC55 progressive-scan camera, which has an extra shutter trigger input with a shutter speed range of 2 μsec to 250 msec. The Galil 1840 card is used for the system's motion control. To improve the efficiency, we capture the image while the SMT machine is still moving. We set the motion speed while capturing the image as 1m/sec and set the shutter speed as 10 μsec. Because the motion card has a sampling time of 250 μsec, we implement another motor-encoder capture card to latch the real time position. The motion card sends a capture signal to the image capturing card's extra shutter trigger input along with the motor-encoder capture card's latch input to capture the image and motor position simultaneously. This capture signal also starts a software interrupt to do the succeeding calculation works. The software is written in Visual C++6.0 and runs on the Windows NT 4.0. The execution time is measured by a PC K6III-400 computer and the total execution time is only 25.5 msec.

As shown in Fig. 8, when the moving sum algorithm is implemented, the maximum position error is reduced about 17~22%. Under a non-uniform illumination circumstance, the improvement is more significant than under a uniform illumination circumstance. The position error's accuracy using a fractional moving sum algorithm is about 0.1 $pixel^2$ in this system. If the position error is too small, then the improvement is saturated as shown in Table 2.

5 Conclusions

We have proposed herein a set of complete procedures for locating and checking the BGA image. In locating the BGA chip, the least squares rectangle algorithm has a faster calculation speed and better robustness than when using the traditional four individual least squares lines method. The moving sum algorithm in the gray frame is used to measure the positions of the BGA pins. This method has a lower sensitivity to image noise and has non-uniform illumination. Finally, the pin's arrangement check is largely simplified by using the CRC algorithm. The proposed methods are verified under the experimental test and their performances are both accurate and rapid. The BGA image system can filter out some noise; and it can also be used under a non-uniform illumination circumstance. The complete locating and checking procedures take only 25.5 milliseconds of execution time when we use a K6III-400 PC computer to place a 348-pin BGA component with an image size of 350x350.

Acknowledgement

This work was support by the National Science Council of Taiwan under Grant NSC 90-2212-E-011-045.

References

[1] G.T. Ayoub, "Machine vision in high-accuracy SMT autoplacers," *Circuits Manufacturing*, Vol. 30, No. 1, pp. 28-32, Jan. 1990.

[2] J. Woolstenhulme and E. Lubofsky, "Machine vision placement considerations," *Surface Mount Technology*, Vol. 14, No. 11, pp. 62-66, 2000.

[3] D.W.Capson and M.C.Tsang, "An experimental vision system for SMD component placement inspection," *IEEE Industrial Electronics Society 16th Annual Conference*, Vol. 1, pp. 815-820, 1990.

[4] C.H. Yeh and D.M. Tsai, "A rotation-invariant and non-referential approach for ball grid array (BGA) substrate conducting path inspection," *International Journal of Advanced Manufacturing Technology*, Vol. 17, No. 6, pp. 412-424, 2001.

[5] Y. Hara, N. Akiyama, and K. Karasaki, "Automatic inspection system for printed circuit boards," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 5, No. 6, pp. 623-630, 1983.

[6] GAW. West, "A system for the automatic visual inspection of bare-printed circuit boards," *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. 14, No. 5, pp. 767-773, 1984.

[7] B. Benhabib, C.R. Charette, K.C. Smith, and A.M. Yip, "Automatic visual inspection of printed circuit boards: an experimental system," International Journal of Robotics and Automation, Vol. 5, No. 2, pp. 49-58, 1990.

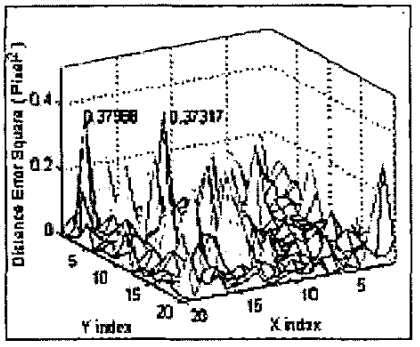
[8] G. Burel, F. Bernard, and W.J. Venema, "Vision feedback for SMD placement using neural networks," *Proceedings of 1995 IEEE International Conference on Robotics and Automation,* Vol. 2, pp. 1491-1496, 1995. [9] J.P. Baartman, A.E. Brennemann, S.J. Buckley, and M.C. Moed. "Placing surface mount components using coarse/fine positioning and vision," *IEEE Trans. on Components Hybrids & Manufacturing Technology*, Vol. 13, No. 3, pp. 559-564, 1990.

[10] S. Hata, K. Hagimae, S. Hibi, and T. Gunji, "Assembled PCB visual inspection machine using image processor with DSP", *IECON'89 15th Annual Conference of IEEE Industrial Electronics*, Vol. 3, pp. 572-577, 1989.

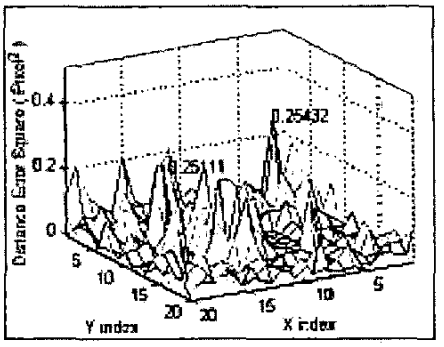
[11] S. Belkasim, A. Ghazal, and O. Basir, "Edge enhanced optimun automatic thresholding," *Proceedings of 2000 ICS:Workshop on Image Processing and Pattern Recognition*, pp.78-85, 2000.

[12] J. Kittler and J. Illingworth, "An automatic threholding algorithm and its performance," *In Proc.*

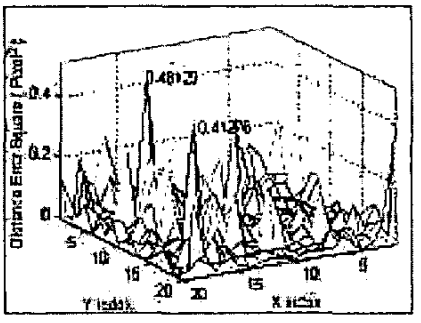
seventh Int. Conference, Pattern recognition, Vol. 1, pp. 287-289, 1984.
[13] W.H. Tsai, "Moment-preserving thresholding : a new approach," *Computer Vision, Graphics & Image Processing*, Vol. 29, No. 3, pp. 377-393, 1985.
[14] T. W. Ridler, S. Calvard, "Picture thresholding using an iterative selection method," *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. 8, No. 8, pp. 630-632, 1978.
[15] P. K. Shaoo, S.Soltani, and A.K.C. Wong, "A survey of thresholding techniques," *Computer Vision, Graphics, Image Processing*, Vol. 41, No. 2, pp. 233-260, 1988.
[16] R. Jain, R. Kasturi and B.G. Schunck, *Machine vision*, McGRAW-HILL,1995.
[17] J.E. Maze and B.R. Saltzberg, "Error-burst detection with randem CRC's," *IEEE Trans. on Communications*, Vol. 39, No. 8, pp.1175-1178, 1991.



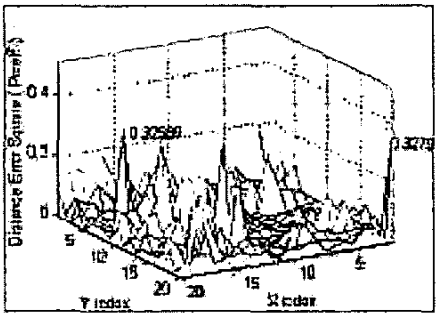
(a) uniform illumination with moving sum



(b) non-uniform illumination with moving sum



(c) uniform illumination without moving sum



(d)non-uniform illumination without moving sum

Figure 8. The experimental results of BGA pin position error.

Table 2. Error reduction percentage

Maximum pins	UNIFORM ILLUMINATION			NON-UNIFORM ILLUMINATION		
	Maximum pin error on average ($pixel^2$)		Error reduction	Maximum pin error on average ($pixel^2$)		Error reduction
	without moving sum	with moving sum		without moving sum	With moving sum	
1	0.461	0.380	17.6 %	0.328	0.254	22.5 %
2	0.437	0.377	13.8 %	0.327	0.253	22.7 %
5	0.380	0.325	14.4 %	0.291	0.246	15.6 %
10	0.312	0.294	5.6 %	0.256	0.236	7.9 %
20	0.254	0.248	2.6 %	0.214	0.210	2.0 %