MVA'90 IAPR Workshop on Machine Vision Applications Nov. 28-30,1990, Tokyo

PRECISE POSITION DETECTION OF A MESH-BACKED PRINTING SCREEN BASED ON THE SMALLEST ENCLOSING CIRCLE DETECTION

Takanori Ninomiya and Yasuo Nakagawa
Production Engineering Research Laboratory
Hitachi, Ltd.

292 Yoshida-cho, Totsuka-ku
Yokohama 244, Japan

ABSTRACT

In this paper we describe a new method of position detection for a mesh-backed printing screen with an alignment mark that is overlapped by mesh patterns. A circular hole is selected as an alignment mark so that at least a portion of its contour can be observed through the windows of the mesh wires. To recognize the true contour, the smallest circle enclosing binary patterns acquired with a transmitted illumination is detected. Then the mark position is calculated by approximating the recognized contour to a circle with the *least square fitting*. To speed up the procedure, a *convex hull* and the *farthest-point Voronoi diagram* is generated before the *smallest enclosing circle* detection. According to the experimental results, an accuracy of 0.2 pixels and a processing time of 1.0 seconds have been achieved.

INTRODUCTION

Determining the position of a specific object in an image is one of the most important and practical applications of digital image processing in industry. Various methods have been developed. For example, Kashioka, *et al.*[1] reported an automatic wire bonding system for transistor chips using the template matching method. The use of the center of gravity or the projection of a detected pattern is also a common method. Discussion on the accuracy issue of detected position is found in a large body of literature;[2][3] etc. Most of the developed methods, however, assume that the pattern to be detected will appear just like it is expected.

In this paper we propose a simple but powerful method for the position detection of a partially hidden pattern, i.e., an alignment mark of a mesh-backed printing screen. A mesh-backed printing screen shown in Fig.1(a) is a mask for printing complicated patterns, such as high-density circuit patterns, on a substrate. To align a screen and a substrate accurately, the positions of both should be precisely determined by directly detecting dedicated alignment marks. As there has been no adequate way to directly detect the positions of mesh-overlapped alignment marks of a screen, alignment marks are printed on a dummy substrate with the screen first, and then an image of the printed marks are detected to determine the screen position.

ALIGNMENT MARKS

A typical shape of an alignment mark consists of straight lines as shown in Fig.1(c). In this case, mesh wires might overlap and hide the whole contour of the alignment mark, because the relative position between a mark and mesh wires cannot be controlled accurately in pattern forming process of a screen. It is impossible to directly detect the position of such a mark with high accuracy.

In our method, a circular hole is adopted as an alignment mark. If the radius r of the circular hole satisfies the following condition, it is guaranteed that at least a portion of its contour is observed through the windows of mesh wires.

$$r > \frac{5\sqrt{2}}{2}d \qquad (1)$$

where d is a diameter of a mesh wire. Moreover, it is probable that the observed contour does not distribute within a semicircle. The binary pattern of the hole observed through the windows can be easily acquired with transmitted illumination, as shown in Fig.2, and with a simple thresholding scheme.

(a) A printing screen (b) An adopted mark (c) A typical mark

Fig.1 A mesh-backed printing screen and its alignment mark

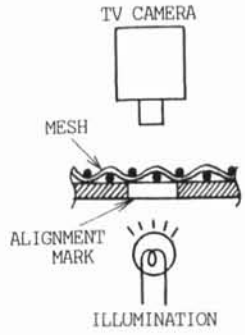


Fig.2 An imaging system

As a result, the smallest circle enclosing the whole of the detected binary pattern inside, referred to as the *smallest enclosing circle*, will agree with the circular alignment mark itself.

ALGORITHM OF POSITION DETERMINATION

Fig.3 shows the flowchart of the algorithm. To achieve a high accuracy of position detection, overcoming digitizing errors, noises and distortion of the imaging system, the final result is calculated by the *least square fitting* of the contour of the alignment mark. The *smallest enclosing circle* is used to extract the true circular contour from the detected binary pattern of a mesh-overlapped alignment mark.

The algorithm consists of the two phases shown in Fig.3: the extraction of a circular contour, and the determination of a circle by the *least square fitting*.

In the following discussion, we assume that the input image is an $m \times m$ pixel binary image with the value "1" for the alignment mark, i.e., a circular hole, observed through mesh windows and the value "0" for another region.

To extract the candidate of a contour of an alignment mark from a set of coordinates of pixels with value "1", the *smallest enclosing circle* is detected and pixels within a certain distance α from the circumference of the *smallest enclosing circle* are extracted.

A brute algorithm for determining the *smallest enclosing circle* is to take all combinations of three "1" pixels, find a circle passing them and check whether all the other "1" pixels are enclosed in the circle and its diameter is minimum. But it requires $O(N^4)$ time and is very time consuming.

To make the procedure efficient in terms of processing time, the proposed method takes the *outermost edges* of the binary pattern, constructs a *convex hull*, and then generates the *farthest-point Voronoi diagram* before determining the *smallest enclosing circle*, as shown in Fig.3. Each step is as follows.

***Outermost edges* and a *convex hull*:** It is obvious that the *smallest enclosing circle* always passes some angular points of a convex polygon enclosing "1" pixels, referred to as a *convex hull*. And also it can be easily proved that if there are more than two points $(x_1,y_0),(x_2,y_0),\ldots,(x_n,y_0)$ $(n>2,\ x_1<x_2<\ldots<x_n)$ with the same y coordinate y_0, points $(x_2,y_0),\ldots,(x_{n-1},y_0)$ cannot be the angular points of a *convex hull* as shown in Fig.4. Therefore one can reduce the amount of input data by extracting the most left-hand "1" pixels and the most right-hand "1" pixels, referred to as *outermost edges*, before construction of a *convex hull*.

We use Graham's algorithm[4] with slight modification for the *convex hull* construction.

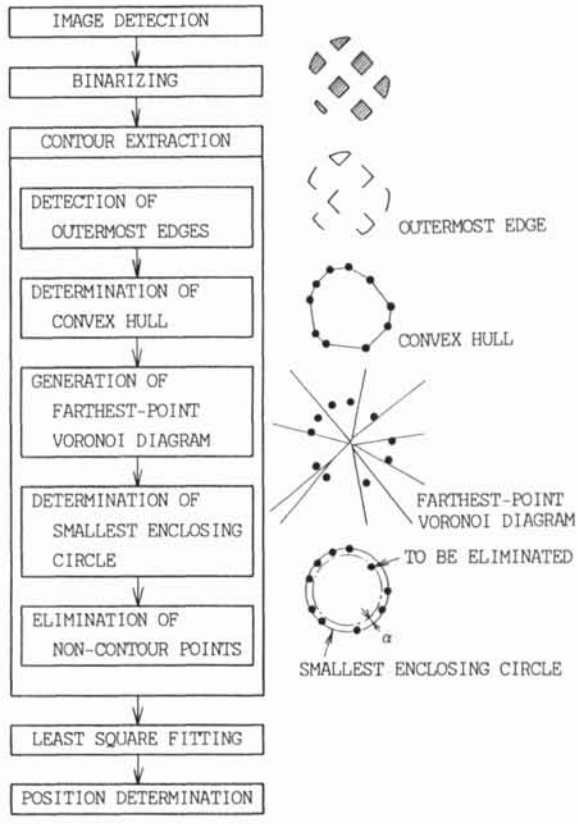


Fig.3 The flowchart of the proposed position detection method

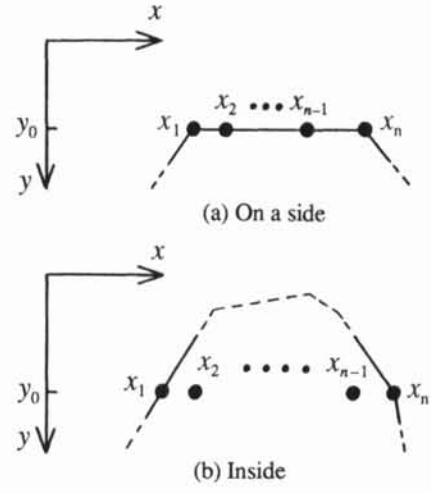
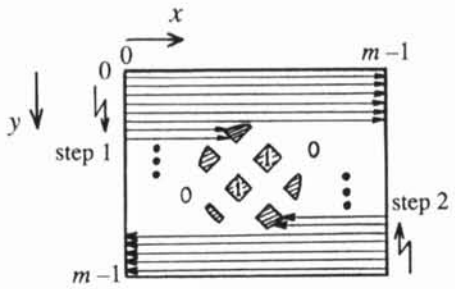
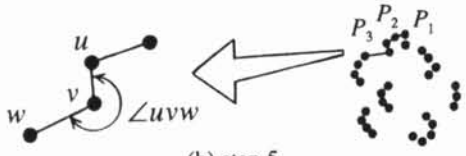


Fig.4 Inner points of a *convex hull*



(a) step 1 and 2



(b) step 5

Fig.5 Construction of a *convex hull*

[*Construction of a convex hull*] (Fig.5)

(step 1) Search the input binary image from the left to the right in each y coordinate and list up the x-y coordinates of the first "1" pixels while increasing y from 0 to $m-1$.

(step 2) Repeat step 1, except that the searching direction is from the right to the left and the listing is performed while decreasing y from $m-1$ to 0.

(step 3) Append the coordinate list of step 2 to the list of step 1, call the total list L and set P_1 to be the point with the minimum x coordinate in the points with the minimum y coordinate in L. Let P_{i+1} be the next data of P_i in L (i=1...).

(step 4) Set $u=P_2$, $v=P_3$, $w=P_4$.

(step 5) Go to step 7 if the angle uvw in Fig.5(b) is greater than or equal to 180 degrees.

(step 6) Set q to be the next data of w in L and set $u=v$, $v=w$ and $w=q$. If $w = P_1$, then stop. Otherwise go to step 5.

(step 7) Set q to be the previous data of u in L, delete v from L, set $v=u$ and $u=q$ and then go to step 5.

This procedure requires $O(N)$ time if the number of the points listed in (step 1) and (step 2) is N.

The *farthest-point Voronoi diagram*: The definition of the *farthest-point Voronoi diagram* is as follows[5].

[*Definition of the farthest-point Voronoi diagram*]

Let $P_1, P_2, \ldots, P_N$ be finite points in the plane, no two of which coincide. The *tile* of P_n is the set T_n defined by

$$T_n = \{z : D(z,P_n) \geq D(z,P_m) \text{ for all } m \neq n\}$$

where D is Euclidean distance. The division of the plane by T_n is called the *farthest-point Voronoi diagram*. The point P_n is called the *generator* of the *tile* T_n.

Fig.6 shows an example of the *farthest-point Voronoi diagram*. From the definition, an intersecting point of three *tiles* comes to the center of a circle which passes the *generators* of these *tiles* and encloses other points. Since the number of intersecting points of *tiles* is at most $O(N^2)$, if the *farthest-point Voronoi diagram* is given, the *smallest enclosing circle* is determined in $O(N^2)$ time.

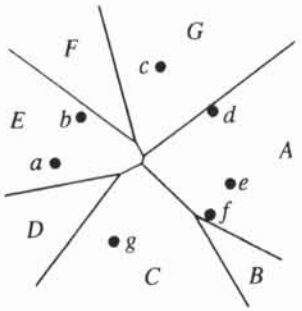


Fig.6 An example of the *farthest-point Voronoi diagram*
Lowercase : *generators*, Uppercase : *tiles*

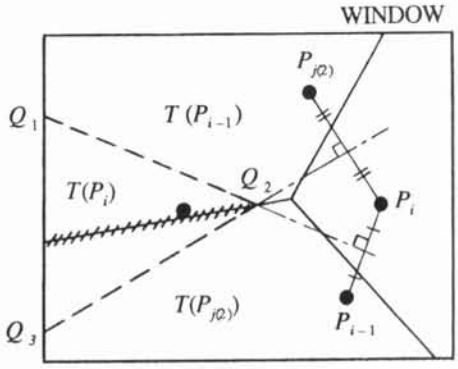


Fig.7 Construction of the *farthest-point Voronoi diagram*

Green and Sibson[5] report an $O(N^2)$ algorithm for constructing the *Voronoi diagram*, while Shamos and Hoey[6] show an $O(NlogN)$ algorithm. We use the Green and Sibson's algorithm (Fig.7).

[*Construction of the farthest-point Voronoi diagram*]

Let $P_1, P_2, \ldots, P_N$ be angular points of the constructed *convex hull*. Divide a processing window by the bisector of a segment P_1P_2 and let the opposite side to P_i be the *tile* $T(P_i)$ (i=1,2). Then execute the following procedure for P_i (i=3,...,N).

(step 1) Set k=1 and $j(k) = i-1$.

(step 2) Let the intersection of the bisector of a segment $P_{j(k)}P_i$ and an edge of the *tile* $T(P_{j(k)})$ of the generator $P_{j(k)}$ be Q_k and let the *tile* neighbouring $T(P_{j(k)})$ at the edge be $T(P_{j(k+1)})$.

(step 3) If $P_{j(k+1)} \neq P_{j(1)}$, set $k=k+1$ and go to step 2. Otherwise let the polygon $Q_1Q_2 \ldots Q_kQ_{k+1}$ be $T(P_i)$ where Q_{k+1} is the intersection of the bisector of a segment $P_{j(k)}P_i$ and an edge of the processing window and stop.

In practice, as shown in Fig.7, the area for the construction is limited to a certain size of a square window.

Extraction of true contour points: The points satisfying the following condition are recognized as the true contour of the circular alignment mark.

$$\left|(x - x_s)^2 + (y - y_s)^2 - r_s^2\right| \leq \alpha^2 \qquad (2)$$

where (x_s, y_s) and r_s are the center and radius of the *smallest enclosing circle* respectively and α is the allowance for extraction.

The *least square fitting*: The approximation of the recognized true contour of a circle by the *least square fitting* can be calculated as follows.

Let the formula of a circle be

$$x^2 + y^2 + 2gx + 2fy + c = 0 \tag{3}$$

and let the coordinates of the true contour points be (x_i,y_i) $(i=1,2,...,N)$. Then g, f and c are determined by solving the following equations.

$$\frac{\partial \varepsilon}{\partial g} = 0\,,\ \frac{\partial \varepsilon}{\partial f} = 0,\ \frac{\partial \varepsilon}{\partial c} = 0 \tag{4}$$

where

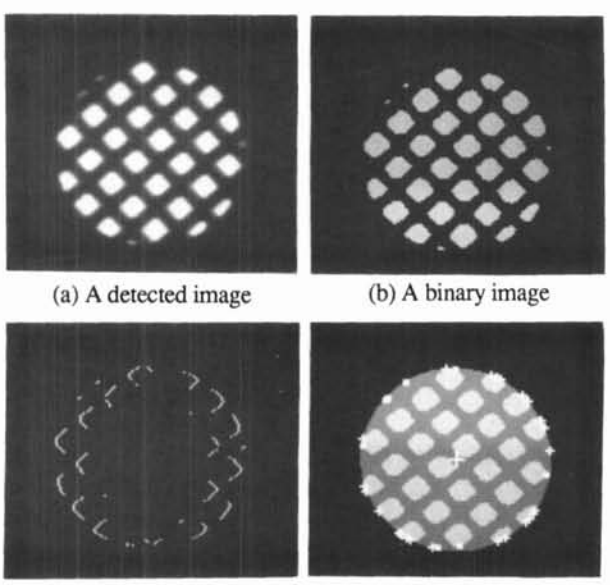
$$\varepsilon = \sum_{i=1}^{N} (x_i^2 + y_i^2 + 2gx_i + 2fy_i + c)^2. \tag{5}$$

The center of the alignment mark is determined as the center of the circle, (–g,–f).

EXPERIMENTAL RESULTS

Fig.8 shows an example of the experimental result. We used a CCD TV camera for image acquisition and took a 256×256 pixel image with 8 bit grayscale. The pixel size was adjusted to about 7 μm square with an optical microscope. In this experiment, the diameter of the alignment mark was 0.6mm, i.e., about 86 pixels. All procedures were done by software on a 1 MIPS microprocessor. We confirmed that the center of an alignment mark was always recognized successfully.

Fig.9 shows the accuracy of the proposed method. An accuracy of 0.2 pixels (1.4μ m) in the worst case was achieved. The processing time was 1.0 second/mark, excluding the *outermost edge* detection. *Outermost edges* can be detected in a short time using simple hardware or a DSP.



(a) A detected image (b) A binary image

(c) Outermost edges (d) A result

Large "+" : The detected position

Fig.8 An example of the position detection

CONCLUSIONS

We have shown a new method for determining the position of a mesh-overlapped alignment mark of a mesh-backed printing screen. The use of a circular mark and the detection of the *smallest enclosing circle* allow the robustness of the algorithm. And by introducing the technique of *computational geometry*, i.e., algorithms for constructing a *convex hull* and the *farthest-point Voronoi diagram*, our method can be executed in practical time with a simple microcomputer system. The proposed method could break one of the bottlenecks in eliminating wasteful dummy printing from the screen-printing process. Also, it may suggest clues to the solution of related problems.

Acknowledgment: The authors wish to thank Dr H. Ebara of Osaka University for his kind suggestion and help in programming our method.

References:

[1]Kashioka,S., Ejiri,M. and Sakamoto,Y.:A transistor wire-bonding system utilizing multiple local pattern matching techniques: IEEE Trans. Syst. Man. & Cybern., **SMC-6**, 8, pp.562-570 (1976)

[2]Gordon,S.J. and Seering,W.P.:Accuracy Issues in Measuring Quantized Images of Straight-Line Features: Proc. of 1986 IEEE Int. Conf. on Robotics and Automation, pp.931-936 (1986)

[3]Havelock,D.I.:Geometric Precision in Noise-Free Digital Images: IEEE Trans. Pattern Analysis & Machine Intelligence, **PAMI-11**, 10, pp.1065-1075 (1989)

[4]Graham,R.L.: An Efficient Algorithm for Determining the Convex Hull of a Finite Planar Set: Information Processing Letters, **1**, pp.132-133 (1972)

[5] Green,P.J. and Sibson,R.:Computing Dirichlet tessellations in the plane: The Computer Journal, **21**, No.2, pp.168-173 (1978)

[6]Shamos,M.I. and Hoey,D.: Closest-Point Problems: Proc. of the 16th IEEE Symp. on Foundations of Computer Science, pp.151-162 (1975)

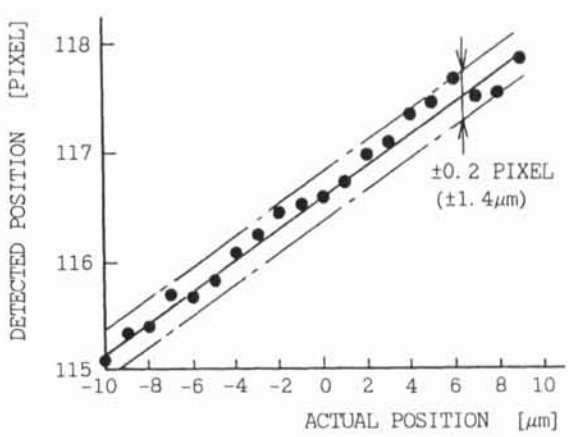


Fig.9 Accuracy of the proposed metod