Point Pattern Matching for 2-D Point Sets with Regular Structure

Tapio Manninen[1], Risto Rönkkä[1], and Heikki Huttunen[2]

[1] DropAim Oy, Tampere, Finland
[2] Tampere University of Technology, Tampere, Finland

Abstract. Point pattern matching (PPM) is a widely studied problem in algorithm research and has numerous applications, e.g., in computer vision. In this paper we focus on a class of brute force PPM algorithms suitable for situations where the state-of-the-art methods do not perform optimally, e.g., due to point sets with regular structure. We discuss of an existing algorithm, which is optimal in the sense of brute force testing of different point pairings. We propose a parameter choosing scheme that minimizes the memory consumption of the algorithm. We also present a modified version of the algorithm to overcome issues related to its implementation and accuracy. Due to its brute force nature, the algorithm is guaranteed to return the best possible result.

Keywords: Point Pattern Matching, Computer Vision, Printed Electronics.

1 Introduction

Point pattern matching (PPM) is a method for finding a one-to-one correspondence and the related transformation between two point sets. What makes finding the right transformation in PPM a non-trivial problem in practice are things like noise in the point locations, unknown correspondence between the points in both of the sets, and possible missing or extra points in either one of the sets. In this paper, we focus on a special PPM case typical of applications in image analysis and computer vision, i.e., finding the optimal similarity transformation (scaling, rotation, and translation) between two sets of points P and Q in $\mathbb{R}^2$.

Good reviews of state-of-the-art PPM algorithms have been written by Li et al. [8] and van Wamelen et al. [14]. Different methods include, e.g., relaxation [11], graph-based approaches [7,5,4,2], iterative approaches [1], and clustering [13,12,3]. The choice of the PPM algorithm depends significantly on the application at hand.

In our earlier papers, we have used point pattern matching as a part of a correction system for printed electronics manufacturing [9]. The problem is to estimate translations of electronic components, and the connectors of the integrated circuits serve as registration points. The related matching problem, however, is slightly different from the usual in that the patterns are man-made and thus regular, as seen in Figure 1. This means that the connectors typically

A. Heyden and F. Kahl (Eds.): SCIA 2011, LNCS 6688, pp. 156–165, 2011.

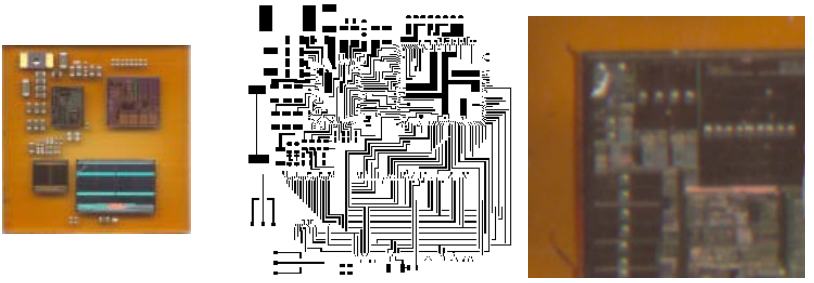

Fig. 1. A sample multi-IC module (left) ready for printing the connections (center). A zoom into the upper left corner is shown in the right.

reside equidistantly on a straight line, which means that several well matching subset pairs can be found. In Section 2, there is a brief description of this kind of application area for PPM.

The most efficient PPM algorithms for the general point pattern matching problem take advantage of the randomness and uniqueness of the patterns in a way that make them unsuitable for our case. Due to the shape of the point sets, the only applicable methods are those using exhaustive search to guarantee the optimal solution even in the presence of good local optima. In this paper, we consider PPM algorithms following the general idea of *alignment* (see Section 3). In the alignment approach, different pairings between the matched point sets are tested to find the best match.

There exists an alignment based PPM algorithm by Chang et al. [3], which will work as a basis for this paper. The algorithm is briefly described in Section 3 where we also show that the Chang's method has an optimal time complexity of $O(m^2n^2)$ among all PPM methods based on the alignment of two sets having m and n points, respectively. However, there are major problems in the algorithm details that can lead to failing of the algorithm or difficulties in the implementation. These problems relate to the two-dimensional accumulator array used in the algorithm to collect scale-rotation pairs created by different alignments of the point sets. The accumulator array is discussed in more detail in Section 4.

In Section 5, we propose a modification of Chang's method that fixes the problems related to the accumulator array. Our solution replaces the array with a search algorithm in continuous space. In Section 6, we discuss more about the practical limitations of the Chang's original algorithm and give an example case where the modified algorithm is necessary. Section 7 concludes the paper.

2 Printed Electronics

The application area, where the need for an efficient point pattern matching method for regular patterns arises is *inkjet printed electronics*. Printed electronics is an additive process and a relatively new area of research, which uses traditional printing devices for interconnecting or manufacturing components.

An example of an application of this technology uses conductive nano particle and dielectric inks to create interconnection circuits between connector pads of integrated circuits (ICs) and discrete components that have been molded onto the background surface [10], see Figure 1.

The problem in the described application is that the embedded components tend to drift away from their intended locations during and after the molding process. In our earlier papers [9], we have proposed a computer vision system to overcome this problem. The processing begins with the acquisition of the image of the distorted module. In order to find the required correction, the connectors of the ICs and their locations are detected from the image by means of automated image analysis. After detection, the problem is to search for a transformation between the two coordinate sets (i.e., to combine the knowledge where the objects are with the knowledge where they should be). The transformation is found using PPM, for which we propose a method in this paper. After the transformation and correspondence between the point sets are available, the wiring is redrawn to match the true situation.

3 Point Pattern Matching

Consider a two-dimensional point set $P = \{\mathbf{p}_k \in \mathbb{R}^2 \mid k = 1, \ldots, m\}$ that is to be matched with the point set $Q = \{\mathbf{q}_k \in \mathbb{R}^2 \mid k = 1, \ldots, n\}$, where $m \leq n$. The aim is to find the parameters of a similarity transformation, i.e., scale $s \in \mathbb{R}^+$, rotation $\theta \in [0, 2\pi)$ of the rotation matrix $\mathbf{R}_\theta$, and translation $\mathbf{t} = (t_x, t_y) \in \mathbb{R}^2$, which maximize the number of matching pairs between sets P and Q. Points $\mathbf{p} \in P$ and $\mathbf{q} \in Q$ are said to match if $||\mathbf{q} - \mathbf{p}'|| < d$, where $d \in \mathbb{R}^+$ is the allowed point distortion and $\mathbf{p}' = s\mathbf{R}_\theta\mathbf{p} + \mathbf{t}$ is the transformed version of $\mathbf{p}$. The resulting transformation should also minimize the sum of the squared errors between the points in Q and the corresponding points in the transformed version of P.

The PPM problem can be divided into two parts. First, we find the best possible pairing of the two points sets P and Q and discard the outlier points in both sets. Second, we find the parameters of the optimal transformation between the paired points. We are only interested in the former problem, because closed form optimal solutions are available for the latter one.

In this paper, we focus on brute force PPM methods based on a general and intuitive idea of *alignment*. In the alignment approach, the point sets are aligned such that points $\mathbf{p}_i$ and $\mathbf{p}_j$ from set P exactly match the points $\mathbf{q}_u$ and $\mathbf{q}_v$ from set Q and then the total number of matching pairs is calculated over the whole sets. In a naive solution, the best match is found by looping each of the $mn(m-1)(n-1)/4$ alignments and calculating the number of matching pairs by finding a nearest neighbor for each point.

Chang's PPM method [3] re-arranges the calculation in the alignment approach such that the number of matching pairs can be calculated in constant time. The best point pairing is found by clustering of scale-rotation pairs achieved by calculating scales and rotations between each possible pair of point pairs $(\mathbf{p}_i, \mathbf{p}_j)$ and $(\mathbf{q}_u, \mathbf{q}_v)$. Pairs with true correspondence tend to cluster around the

same region in the scale-rotation space, while non-matching pairs distribute more randomly. The average scale and rotation of the detected cluster is approximately the scale and rotation of the transformation between the two sets.

After determining the parameters of an approximate transformation that transforms set P into set Q, the actual point correspondence is determined by transforming P with the approximate transformation and pairing each point to the closest point in Q. If no pair is found within a given distance, the point is considered an outlier and discarded. Unlike in the naive case, the nearest neighbors are needed to be found only once.

What makes Chang's algorithm effective is that it uses an accumulator array with size independent of the number of points. The array is used to collect each scale-rotation pair after which the element of the accumulator array with the most hits is found. While efficient, the discrete accumulator array is also the weakest part of the algorithm for several reasons. The main reason is that for point sets with points both relatively close and relatively far from each other, there exists multiple alignment transformations that have very little deviation in their scale and rotation but still result in different point pairing. In this case, the accumulator array has to be very dense in order for the right cluster not to mix with false hits. Unfortunately, large arrays create problems in implementation. In addition, too dense an array increases the possibility of the borders of the array bins to split the target cluster in half, which can make the algorithm to fail. In the original paper, they do not discuss about choosing a proper size for the accumulator. In the next section, we propose a method for choosing the array size in Chang's algorithm such that the total array size is minimized.

4 Choosing Accumulator Array Size for Chang's Method

In the original paper of Chang et al. [3], the authors do not discuss about choosing the dimensions of the accumulator array used for collecting the scale-rotation pairs calculated between each point pair in P and each point pair in Q. However, the size and density of the accumulator array has a significant impact on the performance of the algorithm and it also sets restrictions concerning the implementation of the algorithm.

Like mentioned in the previous section, a single accumulator array bin has to be small enough such that scale-rotation pairs originating from one point set alignment cannot get mixed with the scale-rotation cluster of the alignment that will result from another point pairing. In this section, we will propose a method for determining the largest possible size of the array bin that minimizes the accumulator array memory consumption while preserving adequate resolution for detection of the target cluster.

Consider two points $\mathbf{q}_a$ and $\mathbf{q}_b$ chosen from set Q such that the points are located on distance d_{ab} from each other. Also consider a third point $\mathbf{q}_c \in Q$ with a distance of d_{bc} from point $\mathbf{q}_b$ in direction determined by angle $\phi \in [0, \pi]$. Together with points $\mathbf{p}_i$ and $\mathbf{p}_j$ from set P, we form two different alignments: first, between point pairs $(\mathbf{p}_i, \mathbf{p}_j)$ and $(\mathbf{q}_a, \mathbf{q}_b)$ and, second, between point pairs $(\mathbf{p}_i, \mathbf{p}_j)$ and

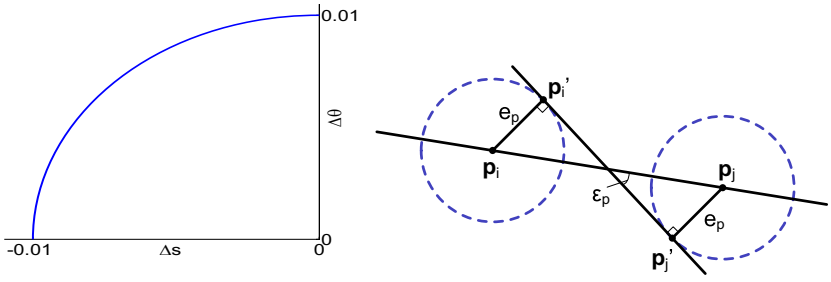

Fig. 2. Left: A curve representing a lower bound for the difference between scales and rotations of two different alignments of sets P and Q. Right: Given the maximum error e_p that the observed points $\mathbf{p}'_i$ and $\mathbf{p}'_j$ can have with respect to their true locations $\mathbf{p}_i$ and $\mathbf{p}_j$, this figure illustrates the maximum error ε_p that can occur in the observed orientation of the point pair.

$(\mathbf{q}_a, \mathbf{q}_c)$. Depending on ϕ, the difference between the scale and rotation of these two alignments can be defined as

$$\Delta s = \frac{d_{ac} - d_{ab}}{d_p} \qquad \text{and} \qquad \Delta\theta = \arccos\left(\frac{d_{ac}^2 - d_{bc}^2 + d_{ab}^2}{2 d_{ac} d_{ab}}\right), \tag{1}$$

where $d_p = ||\mathbf{p}_i - \mathbf{p}_j||$ and $d_{ac} = ||\mathbf{q}_a - \mathbf{q}_c|| = \sqrt{d_{bc}^2 - 2 d_{bc} d_{ab} \cos(\phi) + d_{ab}^2}$. Here we have used the cosine rule.

Let d_{max} and d_{min} be the largest and smallest distances between any two points in Q, respectively. Intuitively, by setting $d_{ab} = d_{max}$ and $d_{bc} = d_{min}$ we get a lower bound for Δs and $\Delta\theta$. Additionally, by denoting $r = d_{min}/d_{max}$, $s = d_{ab}/d_p$, and $r_e = d_{ac}/d_{max} = \sqrt{r^2 - 2r\cos(\phi) + 1}$, we can write these lower bounds for equations (1) as

$$\Delta s = s(r_e - 1) \qquad \text{and} \qquad \Delta\theta = \arccos\left(\frac{r_e^2 - r^2 + 1}{2 r_e}\right), \tag{2}$$

respectively. We restrict further investigation on the interval $\phi \in [0, \phi_{max}]$ where $\phi_{max} = \arccos(r/2)$. This is because the values $\phi \in (\phi_{max}, \pi]$ give $d_{ac} > d_{max}$, which is not desired as d_{max} was chosen as the largest of any two point distances. Equations (2) give us a parametric representation of a lower bound for minimum difference between two different alignments. In Figure 2 (left), we have plotted this lower bound with respect to varying ϕ when $s = 1$ and $r = 1/100$.

The problem is to choose the accumulator array bin dimensions such that two scale-rotation pairs from two different alignments cannot occur in the same bin. Simultaneously, the bin area should be as large as possible implying a small number of bins needed to cover the entire scale-rotation space. An optimization scheme is proposed based on Figure 2 (left): maximize the area of a rectangle (representing an accumulator array bin) confined between the coordinate axis and the (Δs,$\Delta\theta$) curve.

Both Δs and $\Delta\theta$ are monotonically increasing on $\phi \in [0, \phi_{max}]$. As an implication, the maximum area rectangle we are looking for is amongst those rectangles that have one corner in the origin and the opposing corner on the $(\Delta s, \Delta\theta)$ curve. The area of such a rectangle is

$$a(\phi) = |\Delta s||\Delta\theta| = s(1 - r_e) \arccos\left(\frac{1 - r\cos(\phi)}{r_e}\right). \tag{3}$$

The maximization problem of $a(\phi)$ doesn't depend on s. In addition, it isn't necessary to apply any kind of normalization for Δs or $\Delta\theta$ to make sure that we are maximizing a valid type of area because this wouldn't change the location of the optimum. Thus, we can equivalently maximize function

$$J(\phi; r) = \left(1 - \sqrt{r^2 - 2r\cos(\phi) + 1}\right) \arccos\left(\frac{1 - r\cos(\phi)}{\sqrt{r^2 - 2r\cos(\phi) + 1}}\right). \tag{4}$$

Here we have emphasized by substituting r_e from above that there is only one parameter that we need to know beforehand. This parameter is r, i.e., the ratio of the distances between the two closest and two most distant points in Q.

We use numerical methods to solve the maximum point of $J(\phi; r)$ with a given r. After this, we can calculate the accumulator array bin size by using Equations (2). However, Δs still depends on the unknown s, which is the scale of the alignment transformation. This suggests that we shouldn't use a uniform scale axis in our accumulator array. Instead, we need to choose a bin width that increases linearly as the scale gets larger.

The final issue concerning the size of the accumulator array are the truncation points for the scale axis. The optimal scale range for the given point sets P and Q is such that the first scale bin in the array equals to the smallest possible alignment scale $s_{min} = d_{min}/\delta_{max}$ and the last bin equals to the largest possible scale $s_{max} = d_{max}/\delta_{min}$. Here we use δ_{max} and δ_{min} to denote the distance between the two most distant and two closest points in P, similarly to d_{max} and d_{min} that we use for Q. The dimensions of the accumulator array are now determined by the following equations:

$$N_s = \left\lceil \frac{\log(s_{max}/s_{min})}{\log(2 - r_e)} \right\rceil \qquad \text{and} \qquad N_\theta = \left\lceil \frac{2\pi}{|\Delta\theta|} \right\rceil. \tag{5}$$

To achieve N_s we have summed together widths $|\Delta s|$ of N_s bins starting from $s = s_{min}$ and then set this sum equal to $s_{max} - s_{min}$.

To conclude, the entire procedure for choosing the accumulator array size in Chang's PPM method is described in the following steps:

1. Loop through each point pair in P and find δ_{min} and δ_{max}.
2. Loop through each point pair in Q and find d_{min} and d_{max}.
3. Calculate r, ϕ_{max}, s_{min}, and s_{max}.
4. Find $\phi \in [0, \phi_{max}]$ that maximizes $J(\phi; r)$.
5. Divide accumulator rotation axis $[0, 2\pi]$ into bins with equal size of $|\Delta\theta|$.
6. Divide accumulator scale axis $[s_{min}, s_{max}]$ into bins with variable size of $|\Delta s|$.

5 Modified Method without Discrete Accumulator Array

An improved version of the Chang's PPM method [3] is now proposed to overcome the problems related to using a discrete accumulator array. The basic idea is for each scale-rotation pair to individually form a *rectangle* defining the error bounds for that particular scale-rotation pair. The target cluster is then considered as the area where most rectangles intersect each other in the *continuous* scale-rotation space. Dimensions for each rectangle are derived from the given error bounds of each point in the following manner.

Consider point pairs $(\mathbf{p}_i, \mathbf{p}_j)$ and $(\mathbf{q}_u, \mathbf{q}_v)$ from sets P and Q, respectively. Also let $i \neq j$ and $u \neq v$. Assume that, when observed, we can only have the noisy point locations $\mathbf{p}'_i = \mathbf{p}_i + \mathbf{n}_i$, $\mathbf{p}'_j = \mathbf{p}_j + \mathbf{n}_j$, $\mathbf{q}'_u = \mathbf{q}_u + \mathbf{n}_u$, and $\mathbf{q}'_v = \mathbf{q}_v + \mathbf{n}_v$, where $\mathbf{n}_i, \mathbf{n}_j, \mathbf{n}_u, \mathbf{n}_v \in \mathbb{R}$ are random noise terms with arbitrary distributions.

Let's assume that we are provided with constant quantities $e_p, e_q \in \mathbb{R}^+$ that indicate—with a certain confidence level that depends on the distribution of the noise—the maximum error the points in P and Q are assumed to have, respectively. It follows that the distance d_p between the points $\mathbf{p}_i$ and $\mathbf{p}_j$ is now somewhere between the limits $||\mathbf{p}'_i - \mathbf{p}'_j|| - 2e_p \leq d_p \leq ||\mathbf{p}'_i - \mathbf{p}'_j|| + 2e_p$ and, similarly, distance d_q between the points $\mathbf{q}_u$ and $\mathbf{q}_v$ is somewhere between the limits $||\mathbf{q}'_u - \mathbf{q}'_v|| - 2e_q \leq d_q \leq ||\mathbf{q}'_u - \mathbf{q}'_v|| + 2e_q$. From these inequalities it follows that the true scale parameter $s = d_q/d_p$ is somewhere on the interval

$$\frac{||\mathbf{p}'_i - \mathbf{p}'_j|| - 2e_p}{||\mathbf{q}'_u - \mathbf{q}'_v|| + 2e_q} \leq s \leq \frac{||\mathbf{p}'_i - \mathbf{p}'_j|| + 2e_p}{||\mathbf{q}'_u - \mathbf{q}'_v|| - 2e_q}. \tag{6}$$

This gives the worst case error bounds for the scale parameter of the alignment defined by the point pairs $(\mathbf{p}_i, \mathbf{p}_j)$ and $(\mathbf{q}_u, \mathbf{q}_v)$.

Bounds for the rotation parameter θ can be established in a similar manner. Figure 2 (right) illustrates the worst case scenario from the rotation point of view. Both of the measured points $\mathbf{p}'_i$ and $\mathbf{p}'_j$ have the maximum error e_p in their locations. The direction of the error is such that the angle $\varepsilon_p \in [0, \pi/2]$ is maximized. The true orientation θ_p of the line traveling from point $\mathbf{p}_i$ to point $\mathbf{p}_j$, i.e., the angle between the line and the x-axis, is obviously somewhere between the limits $\theta'_p - \varepsilon_p \leq \theta_p \leq \theta'_p + \varepsilon_p$, where θ'_p is the orientation of the line from $\mathbf{p}'_i$ to $\mathbf{p}'_j$ and $\varepsilon_p = \arctan\left(2e_p||\mathbf{p}'_i - \mathbf{p}'_j||^{-1}\right)$. Similarly, for set Q we get $\varepsilon_q = \arctan\left(2e_q||\mathbf{q}'_u - \mathbf{q}'_v||^{-1}\right)$. From these error bounds, we can derive the limits for the true rotation parameter $\theta = \theta_q - \theta_p$:

$$(\theta'_q - \theta'_p) - (\varepsilon_p + \varepsilon_q) \leq \theta \leq (\theta'_q - \theta'_p) + (\varepsilon_p + \varepsilon_q). \tag{7}$$

Given now a scale-rotation pair, instead of accumulating a corresponding bin in a discrete accumulator array, we construct a rectangle, whose dimensions are given by equations 6 and 7. The closer the points $\mathbf{p}'_i$ and $\mathbf{p}'_j$ or the points $\mathbf{q}'_u$ and $\mathbf{q}'_v$ are to each other, the more dominant the error terms e_p and e_q become, thus, resulting in larger rectangles. In calculating ε_p and ε_q we assume that $||\mathbf{p}'_i - \mathbf{p}'_j|| > 2e_p$ and $||\mathbf{q}'_i - \mathbf{q}'_j|| > 2e_q$. This is reasonable as this would otherwise

allow the noise to be so powerful that some of the points in the sets would be able to change places. No point pattern matching method could survive that.

The final step in the improved algorithm is to find the scale-rotation cluster. As the scale-rotation pairs now each form a rectangle indicating its error bounds, the problem—instead of finding the maximum from an accumulator array—becomes finding the area where most rectangles intersect. This can be done by using a sweep line algorithm, familiar, e.g., from computer graphics (*scanline rendering* [15]). The principle of the sweep line algorithm is simple: The scale-rotation space is scanned with a horizontal sweep line that moves in y-direction simultaneously keeping track on rectangles, which intersect the line. The area where most rectangles intersect at the same time is stored in the memory.

Replacing Chang's accumulator array with a sweep line algorithm takes care of the memory issues. However, it also results in increased computational complexity, which we will now discuss a bit more. There are $mn(m-1)(n-1)/4$ different alignments of the two point sets P and Q in the general alignment approach for solving the PPM problem. Thus, the time complexity of any PPM algorithm based on the alignment approach is $O(m^2n^2k)$, where k depends on the complexity of calculating the matching pairs. A naive solution calculates the number of matching pairs by finding the nearest neighbors for each point with a given alignment. Nearest neighbors can be found in $O(m \log n)$ time giving the naive solution a total complexity of $O(m^3n^2 \log n)$.

In Chang's PPM algorithm [3], the number of matching pairs is calculated in constant time. This makes the algorithm optimal in the alignment sense with time complexity of $O(m^2n^2)$. In the improved algorithm proposed in this paper, the optimality is lost as finding the best scale-rotation cluster depends on the number of points. However, there exists an optimal sweep line based solution with time complexity of $O(k \log k)$ for finding the largest possible subset of intersecting rectangles from a set of k rectangles [6]. Thus, the complexity of the improved algorithm is $O(m^2n^2 \log n)$, where the additional logarithmic term is fortunately quite marginal compared to the optimal solution.

6 Example Case

The main problem in Chang's original PPM algorithm is the high memory consumption in cases where the characteristics of the point sets require large accumulator arrays. In Figure 3, we have experimented with the total accumulator array memory consumption with respect to $1/r$, i.e., the ratio between the two most distant and two closest points in Q. To be able to calculate the number of bins in the scale axis for arbitrary point sets, we have assumed $r = \delta_{min}/\delta_{max} = d_{min}/d_{max}$. This results in $s_{max}/s_{min} = r^{-2}$, i.e., the number of scale axis bins (Equation (5)) is independent of s_{min} and s_{max}. In Chang's algorithm, there are at most $m-1$ scale-rotation pairs from a single alignment per iteration. Our accumulator array design guarantees that there cannot be more hits than this in a single array bin. Thus, we have used 16 bit accumulator array bin, which is adequate with all reasonable size point sets.

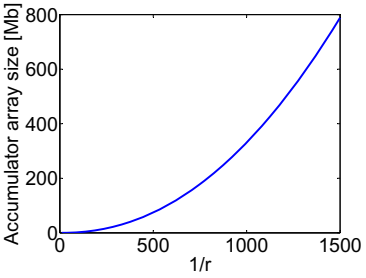

Fig. 3. Memory consumption of a 16 bit accumulator array with respect to the ratio between the distance of the two most distant points and two closest points

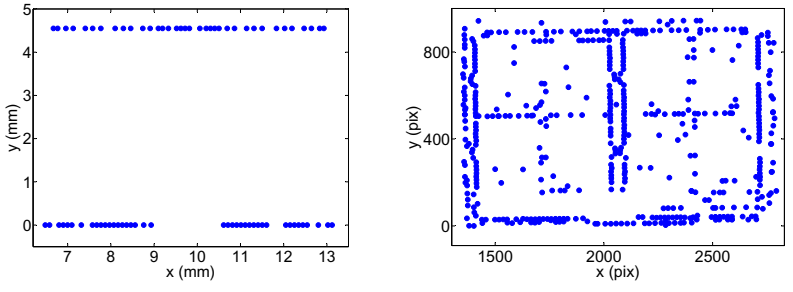

Fig. 4. Left: Locations of IC connector pads according to design data. Right: Locations of connector pad candidates detected from an image.

To provide a real life example, consider the IC registration case presented in Section 2 and the printed electronics module shown in Figure 1. In Figure 4 (left) there are the locations of the connector pads given by the design data of the IC on the lower right corner of the module. Figure 4 (right) shows connector pad candidates of the same IC detected from an image. To be able to find the exact position of the IC in the image, we plan to run Chang's PPM algorithm to find the transformation between the two point sets. However, there exists extremely close points in both of the sets, namely, $d_{min} = 1$ pix, $d_{max} = 1636$ pix, $\delta_{min} = 97.9\,\mu$m, and $\delta_{max} = 7894\,\mu$m. According to Equations (5), we would need an accumulator array of size 27283×14535, which requires 756 MB of memory. Unfortunately, we don't have that much contiguous memory available in the printing lab PC. However, by using the improved algorithm proposed in this paper, we only need to store coordinates of $(m-1)\cdot(n-1)$ rectangles per iteration. In this case this results in extra memory usage of just 4.5 MB.

7 Conclusions

In this paper we have proposed a modified version of the point pattern matching algorithm by Chang et al. [3]. The original method is optimal among all PPM

methods that are based on brute force testing of different alignments, which is the only way to guarantee success in case of regular point sets.

Despite the optimality, Chang's method has some problems, e.g., a high demand for memory when matching point sets with clustered points. We have proposed a method for choosing the accumulator array size that minimizes the memory consumption. In addition, we have proposed an improved version of the original algorithm that overcomes the issues related to using a discrete accumulator array. To the best of our knowledge, these result have not been discovered earlier.

References

1. Besl, P., McKay, H.: A method for registration of 3-D shapes. IEEE Trans. Pattern Anal. Machine Intell. 14(2), 239–256 (1992)
2. Carcassoni, M., Hancock, E.: Point pattern matching with robust spectral correspondence. In: Proc. IEEE Conf. on Comp. Vis. Pattern Recogn. vol. 1, pp. 649–655 (2000)
3. Chang, S., Cheng, F., Hsu, W., Wu, G.: Fast algorithm for point pattern matching: Invariant to translations, rotations and scale changes. Pattern Recogn. 30, 311–320 (1997)
4. Chui, H., Rangarajan, A.: A new point matching algorithm for non-rigid registration. Comp. Vision and Image Understanding 89(2-3), 114–141 (2003)
5. Grimson, W.E.L., Lozano-Pérez, T.: Localizing overlapping parts by searching the interpretation tree. IEEE Trans. Pattern Anal. Machine Intell. 9(4), 469–482 (1987)
6. Imai, H., Asano, T.: Finding the connected components and a maximum clique of an intersection graph of rectangles in the plane. Journal of Algorithms 4(4), 310–323 (1983)
7. Lavine, D., Lambird, B.A., Kanai, L.N.: Recognition of spatial point patterns. Pattern Recogn. 16(3), 289 (1983)
8. Li, B., Meng, Q., Holstein, H.: Point pattern matching and applications-a review. In: IEEE Intern. Conf. on Syst., Man and Cybern., vol. 1, pp. 729–736 (2003)
9. Manninen, T., Pekkanen, V., Rutanen, K., Ruusuvuori, P., Rönkkä, R., Huttunen, H.: Alignment of individually adapted print patterns for ink jet printed electronics. J. Imag. Sci. and Tech. 54(5), 050306 (2010)
10. Miettinen, J., Pekkanen, V., Kaija, K., Mansikkamäki, P., Mäntysalo, J., Mäntysalo, M., Niittynen, J., Pekkanen, J., Saviauk, T., Rönkkä, R.: Inkjet printed system-in-package design and manufacturing. Elsevier Microelectr. J (2008)
11. Ranade, S., Rosenfeld, A.: Point pattern matching by relaxation. Pattern Recogn. 12(4), 269–275 (1980)
12. Stockman, G.: Object recognition and localization via pose clustering. Comp. Vision, Graph. and Image Process. 40(3), 361–387 (1987)
13. Stockman, G., Kopstein, S., Benett, S.: Matching images to models for registration and object detection via clustering. IEEE Trans. Pattern Anal. and Mach. Intell. PAMI-4, 229–241 (1982)
14. Wamelen, P.B.V., Li, Z., Iyengar, S.S.: A fast expected time algorithm for the 2-d point pattern matching problem. Pattern Recogn. 37(8), 1699–1711 (2004)
15. Wylie, C., Romney, G., Evans, D., Erdahl, A.: Half-tone perspective drawings by computer. In: Proc. Fall Joint Comp. Conf., pp. 49–58. ACM, New York (1967)