Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2012, Article ID 139014, 12 pages
doi:10.1155/2012/139014

Research Article

Point Pattern Matching Algorithm for Planar Point Sets under Euclidean Transform

Xiaoyun Wang[1] and Xianquan Zhang[2]

[1] *College of Mathematics and Computer Science, Yangtze Normal University, Fuling, Chongqing 408100, China*
[2] *Department of Computer Science, Guangxi Normal University, Guilin 541004, China*

Correspondence should be addressed to Xiaoyun Wang, wxy5528@163.com

Received 21 December 2011; Revised 2 March 2012; Accepted 3 March 2012

Academic Editor: Tak-Wah Lam

Point pattern matching is an important topic of computer vision and pattern recognition. In this paper, we propose a point pattern matching algorithm for two planar point sets under Euclidean transform. We view a point set as a complete graph, establish the relation between the point set and the complete graph, and solve the point pattern matching problem by finding congruent complete graphs. Experiments are conducted to show the effectiveness and robustness of the proposed algorithm.

1. Introduction

Point pattern matching problem is an important task of computer vision and pattern recognition. It is to find a good correspondence between two point sets in m-dimensional space. In general, it can be classified into two kinds. One is that the point number of the two point sets is the same. The other is that the point numbers are different. The first case is called complete matching, while the second case is called incomplete matching or partial matching. Point pattern matching has found many applications, such as image registration, motion detection, object tracking, automated visual inspection of flat objects, autonavigation, and pose estimation.

In the past decades, many researchers have devoted themselves to designing high performance point pattern matching algorithms. Griffin and Alexopoulos [1] present a method for the complete matching problem. They calculate the two pattern controids and align them and then construct a bipartite graph. The point matching is finally found by determining the maximal cardinality matching of the bipartite graph. Vinod and Ghose [2]

view point pattern matching problem as 0-1 integer programming problem and then use an artificial neural network to solve the point pattern matching problem under translation and rotation transform. To solve the problem of noisy point pattern matching, Morgera and Cheong [3] propose a hybrid and iterative method accommodating patterns from different dimensions. It exploits singular value decomposition to estimate the rotation matrix and steepest-ascent to find the permutation matrix. In [4], Wang and Chen consider the matching problem of point sets with affine transform and use the intrinsic invariant properties of a line segment to design a pattern-matching method. In another work [5], Chang et al. propose a fast algorithm based on 2D cluster approach. It can find the optimal matching between two point sets under the transform of translation, rotation, and scale. In [6], Chui and Rangarajan propose a robust point matching algorithm based on the softassign and the thin-plate spline. This algorithm can jointly estimate the correspondence and nonrigid transformations between two point sets with different sizes. In [7], Carcassoni and Hancock exploit three ways to improve the recovery of point correspondences using spectral analysis of the point proximity matrix. These ways include an alternative proximity weighting matrix, robust methods for comparing the eigenvectors of the proximity matrix and embedding the correspondence process within the EM algorithm. To solve the incomplete matching problem under affine transform, Zhang et al. [8] design a method combining a genetic algorithm with partial Hausdorff distance.

In [9], Van Wamelen et al. propose a fast algorithm for 2D point matching using probabilistic and sorted nearest neighbors. Its time complexity is $O(n(\log m)^{3/2})$, where n and m are the point numbers of the point sets. Li et al. [10] introduce a new similarity K-d tree method to establish a one-to-one match and then apply it to matching nonaffinely-related point sets. To target the optimal transform between point sets, Yin [11] exploits the technique of particle swarm optimization, where the transform parameters are encoded as a real-valued vector called particle. Bishnu et al. [12] present simple and deterministic algorithm for point pattern matching under translation and rotation in 2D. It runs $O(n\,2\log n)$ time for complete matching and $O(mn^{4/3}\log n)$ time for incomplete matching. In [13], Caetano et al. give a graphical models-based algorithm to achieve point pattern matching in Euclidean spaces of any dimension. This algorithm runs in a polynomial time and is provably optimal for complete matching between noiseless point sets. Later, McAuley et al. [14] present a new graph showing better performance than that of [13] and use it to determine point set matching. Li et al. [15] propose a dynamic segment-based hierarchical point matching algorithm for self-initialising articulated motion reconstruction from sparse feature points. Recently, Bhowmick et al. [16] propose a novel data structure called "angular tree" for supporting point pattern matching algorithms. Aiger and Kedem [17] introduce an efficient algorithm for matching point sets under the transform of translation, rotation, and scale by using the Hausdorff metric as a distance function. In another study [18], Aiger and Kedem propose a matching algorithm based on a simple alignment scheme. It runs roughly in $O(n\log n + km\log n)$ time, where m and n are the point number of two point sets and k is the number of matched subsets between the two sets.

Most of the above algorithms are designed to solve the matching problem under affine transform or the transform of translation and rotation. These works are not suitable for the point sets under reflection transform. To overcome this problem, we propose an efficient algorithm based on the fact that the Euclidean distance between any two points is invariant. This algorithm is not only suitable for point sets under reflection transform, but also effective for those under translation and rotation transform. The rest of the paper is organized as follows. Section 2 introduces congruent complete graph-based algorithm. Section 3 presents

the proposed algorithm. Experimental results are given in Section 4 and conclusions are made in Section 5.

2. Congruent Complete Graph-Based Algorithm

2.1. Congruent Complete Graphs

A point set can be represented by a complete graph as follows. A vertex denotes a point and the weight of an edge connecting two vertices represents the Euclidean distance between the corresponding two points. For a given point set, if the point number is n, the edge number of its corresponding complete graph is $n(n-1)/2$. Clearly, if V and V' are two matched point sets under reflection transform, or transform of translation and rotation, the corresponding edges of their complete graphs are equal, and vice versa. In general, translation and rotation transform and reflection transform are collectively called Euclidean transform.

Definition 2.1. If all corresponding edges of two complete graphs are equal, they are congruent graphs.

Let $G(V)$ and $G(V')$ be the complete graphs of V and V', respectively, and $G(V) \cong G(V')$ represent that $G(V)$ and $G(V')$ are congruent graphs. Thus, there is a property of congruent graphs.

Property 1. If $G(V) \cong G(V')$, the point sets V and V' are the matching sets under Euclidean transform.

If two point sets are the matching sets under Euclidean transform, corresponding edges of congruent graphs formed by these point sets are equal, and vice versa. As shown in Figure 1, since $G(\{p_1, p_2, p_3, p_4\}) \cong G(\{q_1,\ q_2,\ q_3,\ q_4\})$, $\{p_1,\ p_2,\ p_3,\ p_4\}$, and $\{q_1, q_2, q_3, q_4\}$ are the matching sets under reflection transform. Similarly, $\{p_1, p_2, p_3, p_4\}$ and $\{v_1, v_2, v_3, v_4\}$ are the matching sets under translation and rotation transform since $G(\{p_1, p_2, p_3, p_4\}) \cong G(\{v_1, v_2, v_3, v_4\})$.

Let V = $v_1, v_2, \ldots, v_k$ $(k \geq 3)$ and $V' = \{v'_1, v'_2, \ldots, v'_k\}$ be two planar point sets. If we use the Euclidean distance between any two points to represent the corresponding edge weight of graph, the complete graph of each point set has $k(k-1)/2$ edges. Clearly, we can find congruent complete graphs by calculating all edges and establishing their corresponding relations. However, such computational cost is large. To reduce computation, we can calculate matched edges based on the following theorem.

Theorem 2.2. *Let $(v_a,\ v'_a)$ and $(v_b,\ v'_b)$ be two pairs of matched points between V and V'. Thus, we have $G(V) \cong G(V')$ if they satisfy the following conditions.*

(1) $|v_a v_i| = |v'_a v'_i|$ $(i = 1, 2, \ldots, k,\ i \neq a)$ *and* $|v_b v_i| = |v'_b v'_i|$ $(i = 1, 2, \ldots, k,\ i \neq b)$.

(2) *If $v_1, v_2, \ldots, v_k$ $(i \neq a, i \neq b)$ are on one side of the straight line $v_a v_b$, their corresponding points $v'_1, v'_2, \ldots, v'_k$ are also on the corresponding side of the straight line $v'_a v'_b$.*

(3) *If v_a, v_b, and v_i $(i \neq a, i \neq b)$ are collinear, their corresponding points v'_a, v'_b, and v'_i are also collinear.*

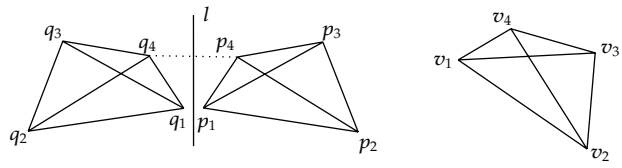
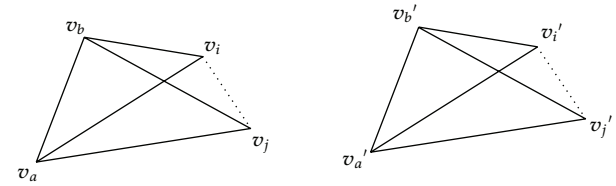
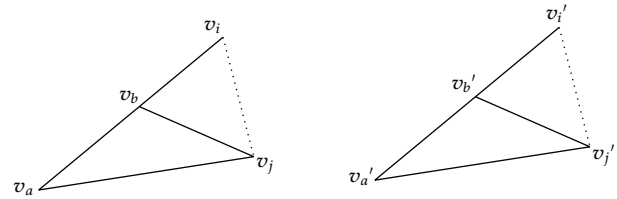


Figure 1: Congruent complete graphs and corresponding point sets.



(a) v_a, v_b, and v_i are not collinear.



(b) v_a, v_b, and v_i are collinear.

Figure 2: Two pairs of congruent complete graphs.

Proof. As shown in Figure 2(a), suppose that v_a, v_b, and v_i $(i \neq a, i \neq b)$ are not collinear, and for a given vertex v_j, there are also no three collinear vertices among $\{v_a, v_b, v_i, v_j\}$. According to the condition (2), assume that v_i and v_j are on one side of the straight line $v_a v_b$. Thus, $v_i{}'$ and v'_j are on the corresponding side of straight line $v'_a v'_b$. Using the condition (1), it is deduced that $|v_a v_b| = |v'_a v'_b|$, $|v_a v_i| = |v'_a v'_i|$, and $|v_b v_i| = |v'_b v'_i|$. Therefore, $G(\{v_a, v_b, v_i\}) \cong G(\{v'_a, v'_b, v'_i\})$ by Definition 2.1. Similarly, $G(\{v_a, v_b, v_j\}) \cong G(\{v'_a, v'_b, v'_j\})$. The Property 1 shows that $\angle v_b v_a v_j = \angle v'_b v'_a v'_j$ and $\angle v_b v_a v_i = \angle v'_b v'_a v'_i$. Thus, $\angle v_i v_a v_j = \angle v'_i v'_a v'_j$ because $\angle v_b v_a v_j = \angle v_b v_a v_i + \angle v_i v_a v_j$ and $\angle v'_b v'_a v'_j = \angle v'_b v'_a v'_i + \angle v'_i v'_a v'_j$. Since $|v_a v_i| = |v'_a v'_i|$ and $|v_a v_j| = |v'_a v'_j|$ (Condition 1), $G(\{v_a, v_i, v_j\}) \cong G(\{v'_a, v'_i, v'_j\})$ by Definition 2.1. Thus, $|v_i v_j| = |v'_i v'_j|$. This means that the distance between v_i and other vertex is equal to that between v'_i and the corresponding vertex. The case that v_i and v_j are on different side of straight line $v_a v_b$ can be proven by using the similar steps. □

Consider the case that vertices are collinear. If v_a, v_b, and v_i are collinear, v'_a, v'_b, and v'_i are also collinear (Condition 3). If v_j and v'_j are on the straight lines $v_a v_b$ and $v'_a v'_b$ respectively, it is clearly that $|v_i v_j| = |v'_i v'_j|$. If they are not on the straight lines as shown in Figure 2(b), $|v_i v_j| = |v'_i v'_j|$ is also available.

From the above analysis, it can be found that $|v_i v_j| = |v'_i v'_j|$ can be derived if the vertices v_i and v_j satisfy the above-mentioned conditions. So $G(V) \cong G(V')$.

Obviously, the pattern matching between two point sets P and Q under Euclidean transform can be achieved by calculating their congruent complete graphs. The detailed algorithm is as follows.

(1) For each vertex $p_i \in P$ $(i = 1, 2, \ldots, n)$, calculate the Euclidean distance $|p_i p_j|$ $(i \neq j)$ between p_i and p_j and sort these distances to make an ascending sequence.

(2) For each vertex $q_i \in Q$ $(i = 1, 2, \ldots, m)$, calculate the Euclidean distance $|q_i q_j|$ $(i \neq j)$ between q_i and q_j and sort these distances to make an ascending sequence.

(3) Compute the congruent complete graphs between $G(P)$ and $G(Q)$ by Theorem 2.2 and determine whether or not P and Q are matched.

2.2. Parameters of Congruent Complete Graphs

If two point sets are matched, one can be viewed as the transformed result of the other, where the transform may be translation and rotation transform or reflection transform. Note that Euclidean distance between two points is invariant under these transforms. Parameters of the above transforms are discussed as follows. (1) Translation and rotation transform denoted as T_{θ,t_x,t_y}: let (x, y) be the coordinates of a point and (x', y') the coordinates of its transformed version. If they satisfy the following equation:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} t_x \\ t_y \end{pmatrix} + \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}, \tag{2.1}$$

the transform is called translation and rotation transform, where θ is the rotation angle t_x and t_y are the translations along x-axis and y-axis, respectively. The θ, t_x, and t_y are the parameters of translation and rotation transform. (2) Reflection transform denoted as T_l: if l is the perpendicular bisector of the line segment connecting (x, y) and (x', y'), the two points are a pair of matched points under the T_l, where l is the symmetry axis and the parameter of the reflection transform. Parameter calculations are as follows.

For translation and rotation transform, let (p_1, q_1) and (p_2, q_2) be two pairs of matched points under translation and rotation transform, that is, $T_{\theta,t_x,t_y}(q_i) = p_i$ $(i = 1, 2)$, where the coordinates of q_i and p_i are (x_i, y_i) and (x_i', y_i'), respectively. Substitute the coordinates of each pair of points into (2.1) and then obtain the following equation:

$$\begin{aligned} t_x &= x_1{}' - x_1 \cos\theta + y_1 \sin\theta, \\ t_y &= y_1{}' - x_1 \sin\theta - y_1 \cos\theta, \\ \theta &= \theta_{\overrightarrow{q_1 q_2}} - \theta_{\overrightarrow{q_1 q_2}}, \end{aligned} \tag{2.2}$$

where θ is the included angle between $\overrightarrow{q_1 q_2}$ and $\overrightarrow{p_1 p_2}$. From (2.2), it is found that the transform parameters t_x, t_y, and θ can be uniquely determined by two pairs of matched points.

For reflection transform, suppose that $\{q_1', q_2', \ldots, q_r'\}$ is the transformed result of $\{q_1, q_2, \ldots, q_r\}$ under the transform T_l, as shown in Figure 3. If q_i $(i = 1, 2, \ldots, r)$ and q_i' are not the same point, the perpendicular bisector l of the line segment connecting q_i and q_i' is the symmetry axis of the reflection transform.

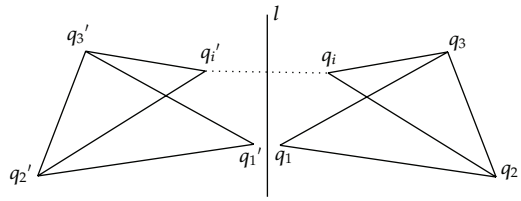


Figure 3: Symmetry axis of reflection transform.

From the above analysis, it is clearly that the parameters t_x, t_y, and θ can uniquely determine a translation and rotation transform, and the symmetry axis l can determine a reflection transform. If $V' = \{v'_1, v'_2, \ldots, v'_k\}$ $(k \geq 3)$ is the transformed result of $V = \{v_1, v_2, \ldots, v_k\}$ under the Euclidean transform, V and V' are a pair of matched point sets.

3. Proposed Algorithm

3.1. Calculation of Congruent Complete Graphs

Let $P = \{p_1, p_2, \ldots, p_m\}$ and $Q = \{q_1, q_2, \ldots, q_n\}$ be two matched point sets in a 2D plane. To find the congruent complete graphs containing vertices p_a and q_b, we calculate the distances $|p_a p_i|$ $(i = 1, 2, \ldots, m,\ i \neq a)$ and $|q_b q_j|$ $(j = 1, 2, \ldots, n,\ j \neq b)$, and make these distances in ascending order, where the sorted result of $|p_a p_i|$ is $\{d_1, d_2, \ldots, d_{m-1}\}$ and that of $|q_b q_j|$ is $\{s_1, s_2, \ldots, s_{n-1}\}$. Thus, congruent complete graphs can be calculated by the following steps.

(1) Let $i = 1$ and $j = 1$. If $d_i = s_j$, record the matched points forming the matched sets and let $i \leftarrow i+1$. If $d_i > s_j$, let $j \leftarrow j+1$. If $d_i < s_j$, let $i \leftarrow i+1$. Repeat the computation until $i = m$ or $j = n$.

(2) Select a pair of matched points p_c and q_f from the above matched sets, whose distances to p_a and q_b are equal. Calculate the distances from p_c and q_f to other corresponding matched points, respectively, and compute the congruent complete graphs by the Theorem 2.2. If the congruent complete graphs are available, we extract other congruent complete graphs from the rest points. Otherwise, we select another matched pair to calculate congruent complete graphs until all matched pairs are used.

3.2. Matching Algorithm Based on Complete Graphs

For incomplete matching problem, if we calculate complete graphs of all points to identify congruent graphs, the computational cost is large. In here, nearest neighbor algorithm is exploited to improve efficiency. Let $P = \{p_1, p_2, \ldots, p_m\}$ and $Q = \{q_1, q_2, \ldots, q_n\}$ be two matched point sets, and the matched probability in P is ρ, that is, ρm points of P have matched points in Q. Thus, the matched probability ρ' in Q can be calculated by

$$\rho' = \frac{\rho m}{n}. \tag{3.1}$$

Note that the minimum point number of congruent complete graphs is 3. Suppose that the selected neighbor point numbers in P and Q are k and t. For each point of P and Q, we apply the method [19] to find its nearest neighbor points. A small number of neighbor points is helpful to improve computational speed. However, average matched point number for each point could not be smaller than 3. In the words, the k or t value must satisfy $\rho k \geq 3$ or $\rho' t \geq 3$. Thus, we have

$$k = \left\lceil \frac{3}{\rho} \right\rceil, \tag{3.2}$$

where $\lceil \cdot \rceil$ means upward rounding. So we have

$$t = \left\lceil \frac{3}{\rho} \right\rceil = \left\lceil \frac{3n}{\rho m} \right\rceil. \tag{3.3}$$

Detailed steps of the proposed matching algorithm are as follows.

Step 1. For each point of Q, extract t neighbor points by the method [19], calculate the distances to the neighbor points, and sort these distances.

Step 2. Randomly select a point from P and find its k neighbor points by the method [19] again. Next, calculate its distances to the k neighbor points, and sort these distances. Apply the Theorem 2.2 to determine matched graphs between the complete graph forming by these $k+1$ points of P and the other complete graph of each point of Q. During the determination, we record the transform parameters and the transform type, that is, translation and rotation transform or reflection transform.

Step 3. Repeat Step 2 and merge the congruent complete graphs with the same transform parameters and the same transform type. If the point number of the congruent complete graphs is bigger than a predefined threshold T, we recalculate the transform parameters. If the calculated results are equal to the prior values, the corresponding points forming the congruent complete graphs are the matched points. The algorithm is done. Otherwise, turn to Step 2.

4. Experimental Results

To validate the proposed algorithm, many experiments are conducted and all results show the effectiveness of our algorithm. In here, typical examples including the synthesized point sets and the real point sets from fingerprints are presented. For the synthesized examples, point sets under the two kinds of Euclidean transform are both considered, where the matched probability in P is 0.8, that is, $\rho = 0.8$. As the points may be perturbed by noises in a real-world situation, we define a threshold $\varepsilon = 4$. If the difference between two distances is smaller than the threshold, their corresponding edges are considered as a matched pair.

Table 1: Coordinates of the points in P.

No.	(x, y)	No.	(x, y)	No.	(x, y)	No.	(x, y)	No.	(x, y)
1	(132, 225)	11	(153, 36)	21	(212, 203)	31	(125, 73)	41	(114, 157)
2	(108, 214)	12	(94, 13)	22	(252, 150)	32	(79, 20)	42	(128, 126)
3	(174, 82)	13	(28, 6)	23	(245, 69)	33	(64, 102)	43	(213, 233)
4	(144, 73)	14	(18, 77)	24	(59, 19)	34	(208, 107)	44	(128, 178)
5	(241, 241)	15	(200, 67)	25	(13, 137)	35	(196, 48)	45	(234, 201)
6	(187, 233)	16	(187, 139)	26	(10, 28)	36	(59, 161)	46	(204, 83)
7	(235, 179)	17	(3, 90)	27	(219, 174)	37	(34, 246)	47	(191, 103)
8	(166, 219)	18	(125, 9)	28	(120, 54)	38	(34, 145)	48	(20, 214)
9	(60, 135)	19	(56, 37)	29	(253, 18)	39	(202, 153)	49	(220, 142)
10	(12, 62)	20	(31, 93)	30	(73, 50)	40	(2, 185)	50	(102, 131)

Table 2: Coordinates of the points in Q under reflection transform.

No.	(x, y)	No.	(x, y)	No.	(x, y)	No.	(x, y)	No.	(x, y)	No.	(x, y)
1	(−63, 110)	11	(−58, 149)	21	(−23, 164)	31	(60, 85)	41	(105, 245)	51	(12, 284)
2	(−94, 165)	12	(−34, 193)	22	(−13, 266)	32	(47, 117)	42	(78, 294)	52	(−8, 204)
3	(−71, 166)	13	(−51, 218)	23	(21, 30)	33	(35, 137)	43	(63, 315)	53	(106, 97)
4	(−102, 196)	14	(−34, 235)	24	(81, −182)	34	(48, 165)	44	(−63, 305)	54	(116, −96)
5	(−61, 205)	15	(−26, 296)	25	(11, 129)	35	(61, 190)	45	(151, 117)	55	(51, −137)
6	(−132, 218)	16	(−7, 31)	26	(4, 238)	36	(163, −69)	46	(163, 149)	56	(93, −49)
7	(−109, 228)	17	(−15, 63)	27	(6, 258)	37	(74, 61)	47	(181, 180)	57	(1, 72)
8	(−87, 245)	18	(−26, 80)	28	(−75, −45)	38	(76, 134)	48	(130, 189)	58	(130, −189)
9	(−41, 88)	19	(1, 93)	29	(25, 313)	39	(98, 149)	49	(163, 199)	59	(52, 342)
10	(−48, 127)	20	(−12, 149)	30	(47, 52)	40	(70, 214)	50	(135, 254)	60	(135, −254)

4.1. Reflection Transform

The point sets are produced as follows. Firstly, 40 random points with integer coordinates are generated in a square sized 256 × 256 to form the point set P. A reflection transform with symmetric axis $y = 2x + 3$ and a translation transform with $t_x = 7$ and $t_y = 4$ are then applied to the 40 points. So 40 points of Q are then obtained. To test the robustness of our algorithm, 10 random points in the 256 × 256 square are generated and then added into P. Another reflection is applied to the 10 points to generate their transformed versions, which are then added to Q. Moreover, another 10 points, which do not belong to P, are added to Q. During the point generations, the distance between each two points is not smaller than 10. Furthermore, the coordinates of points in P and Q are both perturbed by Gaussian noises with 0 mean and 1 variance, and the quantization errors caused by the noises are controlled within ±3. Coordinates of the points in P and Q are presented in Tables 1 and 2, where the serial numbers of points are randomly arranged. As expected, the propose algorithm correctly finds 40 matched pairs of points. The matched results are listed in Table 3.

4.2. Translation and Rotation Transform

The point set P used in Subsection 4.1 is also adopted here. We apply the translation and rotation transform with $t_x = 7$, $t_y = 4$, and $\theta = 0.716$ (radian) to the first 40 points of P and

Table 3: Matched pairs (n_i, n_j) between Tables 1 and 2, where n_i and n_j are the serial numbers of points in Tables 1 and 2.

No.	(n_i, n_j)	No.	(n_i, n_j)	No.	(n_i, n_j)	No.	(n_i, n_j)
1	(11, 11)	11	(26, 23)	21	(36, 39)	31	(22, 8)
2	(46, 13)	12	(39, 27)	22	(44, 40)	32	(12, 9)
3	(34, 14)	13	(27, 51)	23	(43, 43)	33	(3, 12)
4	(22, 15)	14	(45, 29)	24	(25, 53)	34	(4, 21)
5	(13, 16)	15	(10, 30)	25	(40, 45)	35	(19, 57)
6	(24, 17)	16	(20, 31)	26	(48, 46)	36	(16, 26)
7	(32, 18)	17	(33, 32)	27	(37, 47)	37	(5, 59)
8	(30, 19)	18	(50, 34)	28	(18, 1)	38	(9, 38)
9	(31, 20)	19	(41, 35)	29	(15, 5)	39	(1, 41)
10	(49, 22)	20	(17, 37)	30	(29, 6)	40	(6, 42)

Table 4: Coordinates of the points in Q under translation and rotation transform.

No.	(x, y)	No.	(x, y)	No.	(x, y)	No.	(x, y)	No.	(x, y)	No.	(x, y)
1	(−72, 116)	11	(−30, 95)	21	(−4, 303)	31	(69, 75)	41	(107, 200)	51	(37, −206)
2	(−112, 145)	12	(−11, 123)	22	(15, 320)	32	(68, 154)	42	(94, 221)	52	(−94, −81)
3	(−53, 164)	13	(−35, 145)	23	(39, 57)	33	(57, 232)	43	(99, 283)	53	(9, −223)
4	(−117, 179)	14	(−12, 222)	24	(25, 69)	34	(59, 252)	44	(137, 140)	54	(13, −150)
5	(−76, 210)	15	(−40, 260)	25	(29, 90)	35	(80, 256)	45	(115, 145)	55	(101, −45)
6	(−128, 212)	16	(24, 27)	26	(26, 127)	36	(58, 279)	46	(152, 168)	56	(114, 186)
7	(−107, 228)	17	(−3, 32)	27	(−16, −98)	37	(95, 93)	47	(186, 184)	57	(86, −84)
8	(−67, 275)	18	(4, 140)	28	(52, 309)	38	(85, 113)	48	(150, −68)	58	(24, −126)
9	(−24, 59)	19	(−1, 170)	29	(31, 344)	39	(99, 132)	49	(165, 198)	59	(53, 141)
10	(−49, 74)	20	(−9, 197)	30	(54, 71)	40	(85, 180)	50	(147, 217)	60	(−47, −211)

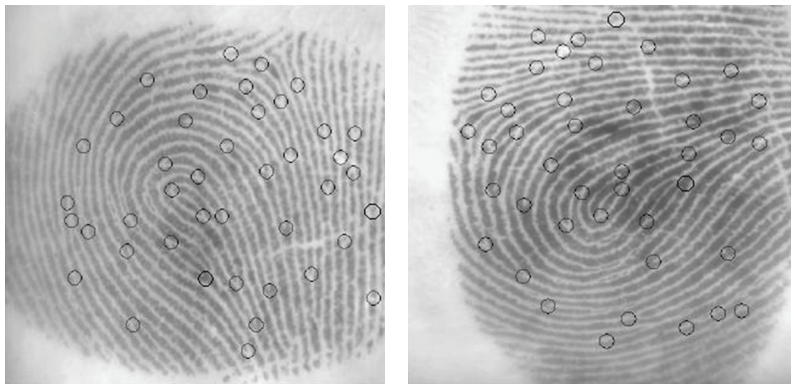
then obtain 40 points of Q. Next, we apply another transform of translation and rotation to the last 10 points of P and add the transformed points to Q. Moreover, another 10 points are added to Q. During generation, the distance between each two points is also not smaller than 10. Similarly, Gaussian noises with 0 mean and 1 variance is also exploited to perturb the points of Q. Table 4 are the coordinates of those points of Q. Our algorithm correctly determines 40 matched pairs of points, as shown in Table 5.

4.3. Fingerprint Matching

To view the performance in a real-world situation, the proposed algorithm is applied to fingerprint recognition. Figure 4 presents two fingerprint images taken from the same finger of a person. The circled feature points are manually selected and their coordinates are also manually extracted by using the software origin. In this experiment, the circled features represent those points for pattern matching and the total number of used features in each fingerprint image is 42. Our algorithm finds 35 matched pairs of features, indicating the practicability of the proposed algorithm. The results are as shown in Figure 5, where the "×" marks represent the feature points in Figure 4(a), the "+" marks represent the feature points in Figure 4(b), and the circle marks mean the matched pairs.

Table 5: Matched pairs (n_i, n_j) between Tables 1 and 4, where n_i and n_j are the serial numbers of points in Tables 1 and 4.

No.	(n_i, n_j)	No.	(n_i, n_j)	No.	(n_i, n_j)	No.	(n_i, n_j)
1	(25, 1)	11	(13, 16)	21	(16, 33)	31	(6, 21)
2	(40, 2)	12	(26, 17)	22	(39, 34)	32	(19, 24)
3	(36, 3)	13	(50, 19)	23	(49, 35)	33	(5, 29)
4	(48, 4)	14	(41, 20)	24	(27, 36)	34	(32, 30)
5	(37, 6)	15	(43, 22)	25	(18, 37)	35	(4, 32)
6	(10, 9)	16	(24, 23)	26	(46, 41)	36	(11, 39)
7	(17, 10)	17	(30, 25)	27	(34, 42)	37	(3, 40)
8	(20, 11)	18	(31, 27)	28	(22, 43)	38	(29, 47)
9	(33, 12)	19	(45, 28)	29	(9, 13)	39	(15, 48)
10	(44, 14)	20	(12, 31)	30	(1, 15)	40	(23, 50)



(a) A fingerprint image (b) Another fingerprint image

Figure 4: Two different fingerprint images from the same finger with circled features.

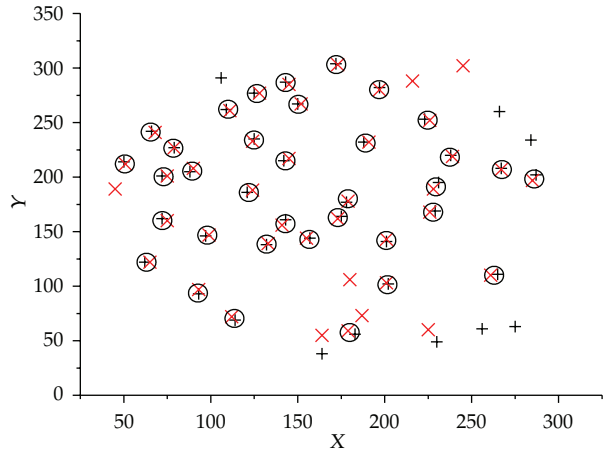


Figure 5: Matched pairs between Figures 4(a) and 4(b).

5. Conclusions

A novel point pattern matching algorithm is proposed in this paper, which views point pattern matching problem as complete graph matching. For each point, the proposed algorithm constructs its complete graph by using its neighbor points. Point pattern matching is then solved by finding congruent complete graphs between the point sets. The proposed algorithm is suitable for the point sets under Euclidean transform. Many experiments are done, and the results show that our algorithm is robust and effective. As finding a matched pair of points in advance is not needed, the proposed algorithm is not influenced by calculation errors causing in point pair determination, and, therefore, achieves robustness. Complete graphs are formed by those points in neighbor. It effectively reduces computational cost and improves speed.

Acknowledgments

This work was partially supported by the Natural Science Foundation of China (60963008), the Guangxi Natural Science Foundation (2011GXNSFD018026, 0832104), the Project of the Education Administration of Guangxi (200911MS55), the Scientific Research and Technological Development Program of Guangxi (10123005–8), and the Scientific and Technological Research Projects of Chongqing's Education Commission (KJ081309). The authors would like to thank Dr. Zhenjun Tang for English correction and the anonymous referees for their valuable comments and suggestions.

References

[1] P. M. Griffin and C. Alexopoulos, "Point pattern matching using centroid bounding," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 19, no. 5, pp. 1274–1275, 1989.

[2] V. V. Vinod and S. Ghose, "Point matching using asymmetric neural networks," *Pattern Recognition*, vol. 26, no. 8, pp. 1207–1214, 1993.

[3] S. D. Morgera and P. L. C. Cheong, "Rigid body constrained noisy point pattern matching," *IEEE Transactions on Image Processing*, vol. 4, no. 5, pp. 630–641, 1995.

[4] W. H. Wang and Y. C. Chen, "Point pattern matching by line segments and labels," *Electronics Letters*, vol. 33, no. 6, pp. 478–479, 1997.

[5] S. H. Chang, F. H. Cheng, W. H. Hsu, and G. Z. Wu, "Fast algorithm for point pattern matching: invariant to translations, rotations and scale changes," *Pattern Recognition*, vol. 30, no. 2, pp. 311–320, 1997.

[6] H. Chui and A. Rangarajan, "New algorithm for non-rigid point matching," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '00)*, vol. 2, pp. 44–51, June 2000.

[7] M. Carcassoni and E. R. Hancock, "Spectral correspondence for point pattern matching," *Pattern Recognition*, vol. 36, no. 1, pp. 193–204, 2003.

[8] L. Zhang, W. Xu, and C. Chang, "Genetic algorithm for affine point pattern matching," *Pattern Recognition Letters*, vol. 24, no. 1–3, pp. 9–19, 2003.

[9] P. B. Van Wamelen, Z. Li, and S. S. Iyengar, "A fast expected time algorithm for the 2-D point pattern matching problem," *Pattern Recognition*, vol. 37, no. 8, pp. 1699–1711, 2004.

[10] B. Li, Q. Meng, and H. Holstein, "Similarity *K*-d tree method for sparse point pattern matching with underlying non-rigidity," *Pattern Recognition*, vol. 38, no. 12, pp. 2391–2399, 2005.

[11] P. Y. Yin, "Particle swarm optimization for point pattern matching," *Journal of Visual Communication and Image Representation*, vol. 17, no. 1, pp. 143–162, 2006.

[12] A. Bishnu, S. Das, S. C. Nandy, and B. B. Bhattacharya, "Simple algorithms for partial point set pattern matching under rigid motion," *Pattern Recognition*, vol. 39, no. 9, pp. 1662–1671, 2006.

[13] T. S. Caetano, T. Caelli, D. Schuurmans, and D. A. C. Barone, "Graphical models and point pattern matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 10, pp. 1646–1662, 2006.

[14] J. J. McAuley, T. S. Caetano, and M. S. Barbosa, "Graph rigidity, cyclic belief propagation, and point pattern matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 11, pp. 2047–2054, 2008.

[15] B. Li, Q. Meng, and H. Holstein, "Articulated motion reconstruction from feature points," *Pattern Recognition*, vol. 41, no. 1, pp. 418–431, 2008.

[16] P. Bhowmick, R. K. Pradhan, and B. B. Bhattacharya, "Approximate matching of digital point sets using a novel angular tree," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 769–782, 2009.

[17] D. Aiger and K. Kedem, "Geometric pattern matching for point sets in the plane under similarity transformations," *Information Processing Letters*, vol. 109, no. 16, pp. 935–940, 2009.

[18] D. Aiger and K. Kedem, "Approximate input sensitive algorithms for point pattern matching," *Pattern Recognition*, vol. 43, no. 1, pp. 153–159, 2010.

[19] A. G. Percus and O. C. Martin, "Scaling universalities of kth-nearest neighbor distances on closed manifolds," *Advances in Applied Mathematics*, vol. 21, no. 3, pp. 424–436, 1998.