グラフィクスとＣＡＤ　66−14
(1993. 12. 17)

文字認識を目的とした１次元ー２次元ＤＰマッチングの最適化

迫江 博昭　Muhammad Masroor Ali　片山 喜規

九州大学工学部情報工学科

〒８１２ 福岡市東区箱崎六丁目１０番１号

あらまし　入力パターンを２次元ドットパターンとし、標準パターンを方向ベクトルの時系列として構造解析的パターンマッチングを Dynamic Programming (DP) で行なう、１次元−２次元 DP マッチングに関して検討した。迫江による Rubber stiring matching を出発点として、線分方向に一定範囲内の適応性を持たせることにより、マッチングの柔軟性を向上した。線分方向の適応範囲 ±４５°, ±約２０° の場合に関して具体的アルゴリズムを示した。元の Rubber string matching（±０° に相当）を含めて比較実験を行ない、±約２０°が最適との結果を得た。

和文キーワード　文字認識 ダイナミック・プログラミング RS マッチング パターン認識 構造解析

One Dimensional–Two Dimensional Dynamic Programming Matching Algorithm Optimization for Character Recognition

Hiroaki Sakoe, Muhammad Masroor Ali, and Yoshinori Katayama

Department of Computer Science and Communication Engineering, Kyushu University

Hakozaki 6-10-1, Higashi-ku, Fukuoka, 812 JAPAN

Abstract

Dynamic programming based elastic pattern matching methods were investigated. In these methods, the reference pattern is represented as a sequence of direction specified vectors and the input pattern as two dimensional dot pattern. Starting from Sakoe's Rubber string matching, adaptation techniques for the reference pattern were newly investigated, improving the flexibility of matching. Experimental results show the effectiveness of the proposed algorithm.

英文 key words　Character recognition　Dynamic programming　Rubber string matching
Pattern matching　Structural analysis

1 Introduction

Recognition of characters have attracted the researchers for more than three decades. And a large number of character recognition systems have been developed so far.

In those systems, pattern matching method is one of the major principles used for recognition. This method is applied mainly to recognition of machine–printed character, since direct application of the method to hand–written character is improper due to wide variation of character patterns.

Sakoe [1], [2] proposed a dynamic programming (DP) [3] based pattern matching algorithm applicable to hand–written character recognition. It conducts an adaptive matching between a directional vector sequence (one dimensional) reference pattern and a dot matrix (two dimensional) input pattern. The algorithm, called rubber string matching (RSM) also possesses a structural analysis feature. It renders an explanation for each black point in input pattern as a correspondence to the reference line segment. Kovalevsky [4] is another researcher who reported DP based pattern matching method. He did not, however, report sufficient description of the algorithm. We can find only several lines describing the basic concept of the algorithm and a figure of experimental results applied to machine printed character.

In this paper, starting from Sakoe's RSM[2], we investigate the optimization of the DP based one dimensional to two dimensional pattern matching algorithm. An adaptation technique for the reference direction were newly proposed improving the flexibility of matching. Practical algorithms which allow ±45° and approximately ±20° adaptation were shown in section 3. In section 4, the character recognition results of the original RSM and the new versions were mutually compared. The results show that the approximately ±20° version gives the best recognition score.

2 The Principle of 1Dim–2Dim Matching

2.1 General Discussion

A character pattern can be approximated by an ordered set of straight line segments each with certain length and direction. For example, the character '2' in Fig. 1 can be approximated by the seven connected straight lines $(C_1, C_2, \ldots, C_7)$. As a general rule, a character consisting of M straight line segments can be represented as,

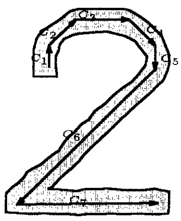


Fig. 1: Approximation of character pattern by directional vectors.

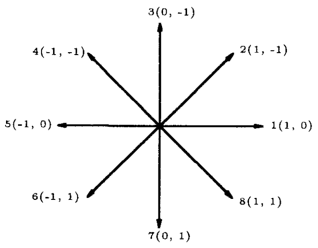


Fig. 2: The eight directional vectors used for character pattern approximation.

$$C = \{(C_1, C_2, \ldots, C_k \ldots C_M), (i_M, j_M)\}, \tag{1}$$

where, (i_M, j_M) are the end point coordinates of the last segment M, designating the location of the character.

Segment C_k with direction d_k and length l_k can be expressed as,

$$C_k = (d_k, l_k). \tag{2}$$

The direction d_k can be one of the eight directions shown in Fig. 2,

$$d_k = (\Delta i_k, \Delta j_k). \tag{3}$$

For the reference pattern, the sequence of d_k's will be stored in the memory. By incorporating parameters l_k and (i_M, j_M) to it, the reference pattern is transformed into two dimensional line patterns. These line patterns are compared with the input patterns to calculate the degree of similarity. The parameters l_k $(k = 1 \sim M)$ and (i_M, j_M) are adjusted for maximum similarity, and this maximum value gives the degree of similarity. At this maximum value the reference pattern is optimally approximated to the input pattern, which is called the approximated skeleton pattern or simply skeleton pattern. If this general rule is applied to all the possible numerous combinations of l_k ($k = 1 \sim M$) and (i_M, j_M), the computations involved become prohibitively enormous. This is why the utilization of DP technique becomes necessary.

2.2 RS Matching

The input pattern is represented by the degree of darkness in a two dimensional mesh ($I \times J$),

$$A = a(i,j); 1 \leq i \leq I, 1 \leq j \leq J. \tag{4}$$

The reference pattern will be thought as a collection of b_k's,

$$B = b_1, b_2, \ldots, b_k, \ldots, b_M, \tag{5}$$

$$b_k = (\Delta i_k, \Delta j_k, w_k, m_k, n_k), \tag{6}$$

where

$d_k = (\Delta i_k, \Delta j_k)$ direction of line segment k.

w_k weight factor,

the degree of darkness or significance.

m_k, n_k permissible range of the length l_k,

$$m_k \leq l_k \leq n_k. \tag{7}$$

Segment k, with end coordinates (i_k, j_k) and length l_k will include l_k points (meshes) given by,

$$\begin{gathered}(i_k - \Delta i_k (x-1), j_k - \Delta j_k (x-1)), \\ x = 1 \sim l_k.\end{gathered} \tag{8}$$

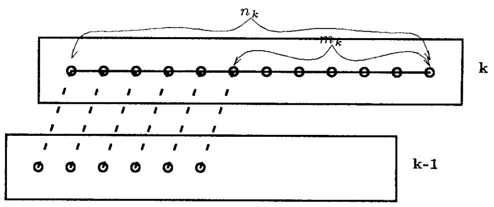
If the level of darkness in the above points is taken as w_k, and in the rest of the points as 0, scalar product with the input pattern of Eq. 4 gives,

$$h_k(i_k, j_k, l_k) = \sum_{x=1}^{l_k} w_k a(i_k - \Delta i_k (x-1), j_k - \Delta j_k (x-1)). \tag{9}$$

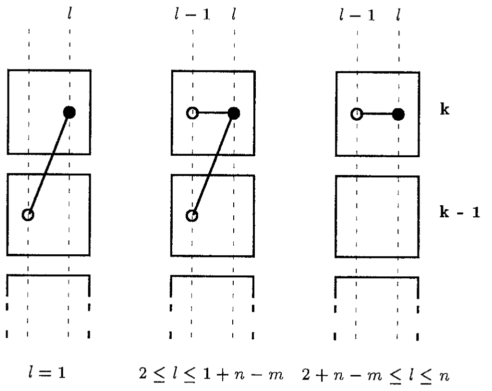
Therefore, the degree of matching (scalar product) for the whole character will be the sum of h_k's, i.e.,

$$\sum_{k=1}^{M} h_k(i_k, j_k, l_k). \tag{10}$$

As explained before, since the maximum value of similarity is defined as the degree of similarity, the same



(a) Basic operation in RS matching.



(b) Unit length transitions in IRS matching algorithm.

Fig. 3: Basic operations in RSM and IRSM.

for the input pattern A and reference pattern B will be expressed as,

$$S(A,B) = \max_{\substack{(l_1, l_2, \ldots, l_M) \\ (i_M, j_M)}} \left[\sum_{k=1}^{M} h_k(i_k, j_k, l_k) \right]. \tag{11}$$

Since the expression within square brackets is in the summation form, we can apply DP for finding the maximum value. By applying DP, we form the recurrence equations shown below.

Initial condition ($k = 0$),

$$g(i,j,0) = 0, \quad 1 \leq i \leq I, 1 \leq j \leq J. \tag{12}$$

Recurrence equation ($k = 1, 2, \ldots, M$),

$$\begin{gathered} g(i,j,k) = \\ \max_{m_k \leq l \leq n_k} [h(i,j,l) + g(i - \Delta i_k l, j - \Delta j_k l, k-1)], \\ 1 \leq i \leq I, 1 \leq j \leq J. \end{gathered} \tag{13}$$

Similarity measure,

$$S(A,B) = \max_{(i,j)} [g(i,j,M)]. \tag{14}$$

The recurrence operation carried out by Eq. 13 is shown in Fig. 3(a).

2.3 Incremental Rubber String Matching Algorithm

Based on the above, we are going to develop more flexible rubber string matching algorithm. As a first step, we break down the operation shown in Fig. 3(a) in terms of unit length transitions shown in Fig. 3(b) to get the incremental rubber string (abbreviated to IRS) matching algorithm. Each transition is along one of the eight vectors of Fig. 2. The unit length transitions will be performed on the two working spaces,

$$g(i,j,k) \quad k = 1,2,\cdots M,$$
$$g_g(i,j,l) \quad l = 1,2,\cdots n.$$

Based on the aforementioned unit length transitions, we form the relations shown below.
Initial condition (k = 0),

$$g(i,j,0) = 0, 1 \le i \le I, 1 \le j \le J \qquad (15)$$

Recurrence relations, for $k = 1,\cdots,M$,

1. $g_g(i,j,l) = g(i-\Delta i_k, j-\Delta j_k, k-1) + a(i,j)$,
   $l = 1,\ 1 \le i \le I,\ 1 \le j \le J.$

2. $g_g(i,j,l) =$
   $\max\begin{bmatrix} g(i-\Delta i_k, j-\Delta j_k, k-1) \\ g_g(i-\Delta i_k, j-\Delta j_k, l-1) \end{bmatrix} + a(i,j),$
   $2 \le l \le 1+n-m,\ 1 \le i \le I,\ 1 \le j \le J.$

3. $g_g(i,j,l) = g_g(i-\Delta i_k, j-\Delta j_k, l-1) + a(i,j)$,
   $2+n-m \le l \le n,\ 1 \le i \le I,\ 1 \le j \le J.$

4. $g(i,j,k) = g_g(i,j,n)$,
   $1 \le i \le I,\ 1 \le j \le J. \qquad (16)$

It is to be noted that with respect to performance, RS and IRS are equivalent.

Practically, it is not necessary to perform the above recurrence calculations for the whole *i*–*j* surface. Since each line segment in a character generally lies in a certain area, we assign a rectangular region i_k^1, j_k^1, i_k^2, j_k^2 to each b_k, where (i_k^1, j_k^1) and (i_k^2, j_k^2) are the coordinates of the top–left and bottom–right corners respectively of the rectangle.

The problem with IRS matching is that it is too rigid with respect to the line direction, and thus may fail to deal with abrupt aberrations of the input pattern points, especially when the input line is thin. To deal with this limitation, we pass on to the flexible rubber string matching methods in the next section.

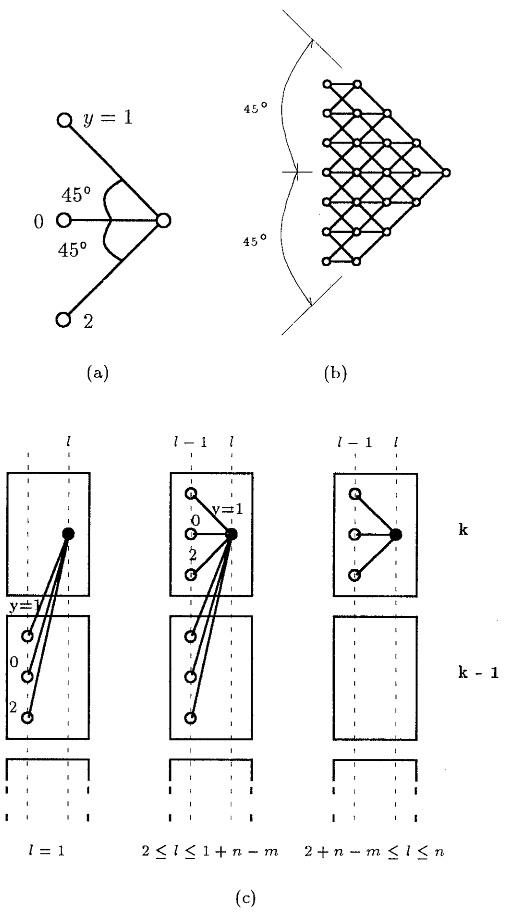


Fig. 4: For 45° FRS matching algorithm (a) Basic transition, (b) Possible course of a stroke with flexibility applied, (c) Unit length transitions.

3 Flexible Matching Methods

3.1 45° Flexible Rubber String Matching Algorithm

In the flexible matching algorithm, instead of a single vector, we are going to consider three consecutive vectors from Fig. 2 as shown in Fig. 4(a). From now on, we are going to call the middle one the main vector and the other two at 45° angle with it the adjacent vectors. Subscript *y* will be used to distinguish the various parameters related to the vectors, with $y = 0$ for the main vector and $y = 1$ or 2 for the adjacent vectors. Thus the slope of the main vector for reference stroke k, so far designated as $(\Delta i_k, \Delta j_k)$, is $(\Delta i_{k_0}, \Delta j_{k_0})$, and those for the adjacent vectors are $(\Delta i_{k_1}, \Delta j_{k_1})$ and $(\Delta i_{k_2}, \Delta j_{k_2})$.

The path in a stroke with this flexibility applied will be as shown in Fig. 4(b). A little observation of the figure will reveal that the outmost paths make an angle of 45° with the path along the main vector, and hence the name 45° FRS matching. When we break down the course in terms of unit length transitions, the situation will be as shown in Fig. 4(c). Each transition may have one of the three directions, one along the main vector and the other two along the adjacent vectors. From now on, we are going to designate the transitions along the main vector as the main transition, and those along the adjacent vectors as adjacent transitions.

A priority coefficient will be used to emphasize any type of transition by multiplying the value for each of the correspondences with the coefficient. For example, the priority coefficient p_y used in the set of relations developed below, may be of the form (3, 2, 2) for $y = 0, 1, 2$.

As before, the unit length transitions will be performed on the two working spaces,

$$\begin{array}{ll} g(i,j,k) & k = 1, 2, \cdots M, \\ g_g(i,j,l) & l = 1, 2, \cdots n. \end{array}$$

Based on the transitions, we form the relations shown below.

Initial condition (k = 0),

$$g(i,j,0) = 0, 1 \leq i \leq I, 1 \leq j \leq J. \qquad (17)$$

Recurrence relations, for $k = 1, \cdots, M$,

1. $g_g(i,j,l) =$
$$\max_{y=0,1,2} \left[g(i - \Delta i_{k_y}, j - \Delta j_{k_y}, k-1) + p_y a(i,j) \right],$$
$$1 = 1, 1 \leq i \leq I, 1 \leq j \leq J.$$

2. $g_g(i,j,l) =$
$$\max_{y=0,1,2} \begin{bmatrix} g(i - \Delta i_{k_y}, j - \Delta j_{k_y}, k-1) + p_y a(i,j) \\ g_g(i - \Delta i_{k_y}, j - \Delta j_{k_y}, l-1) + p_y a(i,j) \end{bmatrix},$$
$$2 \leq l \leq 1 + n - m, 1 \leq i \leq I, 1 \leq j \leq J.$$

3. $g_g(i,j,1) =$
$$\max_{y=0,1,2} \left[g_g(i - \Delta i_{k_y}, j - \Delta j_{k_y}, l-1) + p_y a(i,j) \right],$$
$$2 + n - m \leq l \leq n, 1 \leq i \leq I, 1 \leq j \leq J.$$

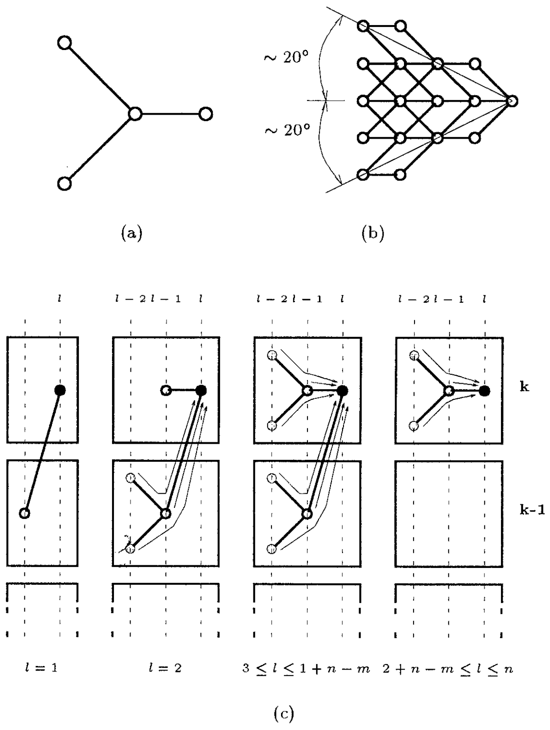


Fig. 5: For 20° FRS matching algorithm (a) Basic correspondence, (b) Possible course of a stroke with flexibility and slope constraint applied, (c) Unit length transitions.

4. $g(i,j,k) = g_g(i,j,n),$
$$1 \leq i \leq I, 1 \leq j \leq J. \qquad (18)$$

As in IRS matching, regions will be assigned to each reference pattern stroke.

3.2 20° Flexible Rubber String Matching Algorithm

While the 45° rubber string matching method is good for dealing with abrupt detours in the input pattern strokes, the problem with it is that it is too flexible. Sakoe and Chiba[5] proposed 'Neither too steep nor too gentle a gradient' for their warping function to be used in DP based speech recognition. Taking this slope constraint method as a base, we propose a flexible rubber string method which is somewhat midway of the previous two methods discussed so far, namely the IRS and 45° FRS matching methods. Here, though we still consider three vectors i. e. the main vector and the adjacent vecotors for a stroke, we impose the constraint that no

Algorithm	Priority coefficient	Recognition Percentage										
		0	1	2	3	4	5	6	7	8	9	Average
IRS	—	98.99	98.48	93.93	98.63	98.48	98.70	98.34	99.64	96.10	97.90	97.92
45° FRS	(1,1,1)	96.38	99.93	98.63	84.38	95.08	80.48	80.19	94.87	90.96	93.35	91.43
	(3,2,2)	99.93	98.84	98.41	99.93	99.35	98.84	98.84	95.95	99.86	99.64	98.96
	(2,1,1)	100.00	98.84	98.77	100.00	99.71	98.92	98.70	94.94	99.93	99.35	98.92
20° FRS	(1,1,1)	99.86	100.00	99.49	99.93	99.71	98.92	100.00	98.99	97.11	99.78	99.38
	(3,2,2)	99.93	100.00	99.42	100.00	100.00	99.20	100.00	99.78	99.86	99.93	99.81
	(2,1,1)	99.86	100.00	99.20	99.71	99.86	99.20	99.86	98.63	99.78	99.71	99.58

Table 1: Recognition accuracy rates for IRS(= RSM), 45° FRS and 20° FRS matching. Group of numbers under the heading 'Priority coefficient' are the coefficient values for main and adjacent vectors. For example, priority coefficient (3, 2, 2) means $p_0 = 3$, $p_1 = 2$, $p_2 = 2$.

two consecutive transitions with a reference stroke can be along adjacent vectors. Thus a possible path with flexibility applied and this constraint imposed will be as shown in Fig. 5(a). And when all these are concatenated, it will be as shown in Fig. 5(b). From the figure we note that the line along the outmost path of the concatenated course diagram makes an angle of approximately 20° with the path along the main vector. When we break down the course in terms of unit length transitions, the situation will be as shown in Fig. 5(c). As in 45° FRS, priority coefficients will be used for emphasizing transitions.

As before, the unit length transitions will be performed on the two working spaces,

$$\begin{array}{ll} g(i,j,k) & k = 1, 2, \cdots M, \\ g_g(i,j,l) & l = 1, 2, \cdots n. \end{array}$$

Based on the transitions, we form the relations shown below.

Initial condition (k = 0),

$$g(i,j,0) = 0, 1 \le i \le I, 1 \le j \le J. \qquad (19)$$

Recurrence relations, for $k = 1, \cdots, M$,

1. $g_g(i,j,l) =$
$$g(i - \Delta i_{k0}, j - \Delta j_{k0}, k-1) + p_0 a(i,j),$$
$$l = 1,\ 1 \le i \le I,\ 1 \le j \le J.$$

2. $g_g(i,j,l) =$
$$\max \begin{bmatrix} g(i - \Delta i_{k0}, j - \Delta j_{k0}, k-1) + p_0 a(i,j) \\ g(i - \Delta i_{k0} - \Delta i_{k1}, j - \Delta j_{k0} - \Delta j_{k1}, k-1) + \\ p_1 a(i - \Delta i_{k0}, j - \Delta j_{k0}) + p_1 a(i,j) \\ g(i - \Delta i_{k0} - \Delta i_{k2}, j - \Delta j_{k0} - \Delta j_{k2}, k-1) + \\ p_2 a(i - \Delta i_{k0}, j - \Delta j_{k0}) + p_2 a(i,j) \\ g_g(i - \Delta i_{k0}, j - \Delta j_{k0}, l-1) + p_0 a(i,j) \end{bmatrix},$$
$$l = 2,\ 1 \le i \le I,\ 1 \le j \le J.$$

3. $g(_g(i,j,l) =$
$$\max \begin{bmatrix} g(i - \Delta i_{k0}, j - \Delta j_{k0}, k-1) + p_0 a(i,j) \\ g(i - \Delta i_{k0} - \Delta i_{k1}, j - \Delta j_{k0} - \Delta j_{k1}, k-1) + \\ p_1 a(i - \Delta i_{k0}, j - \Delta j_{k0}) + p_1 a(i,j) \\ g(i - \Delta i_{k0} - \Delta i_{k2}, j - \Delta j_{k0} - \Delta j_{k2}, k-1) + \\ p_2 a(i - \Delta i_{k0}, j - \Delta j_{k0}) + p_2 a(i,j) \\ g_g(i - \Delta i_{k0}, j - \Delta j_{k0}, l-1) + p_0 a(i,j) \\ g_g(i - \Delta i_{k0} - \Delta i_{k1}, j - \Delta j_{k0} - \Delta j_{k1}, l-2) + \\ p_1 a(i - \Delta i_{k0}, j - \Delta j_{k0}) + p_1 a(i,j) \\ g_g(i - \Delta i_{k0} - \Delta i_{k2}, j - \Delta j_{k0} - \Delta j_{k2}, l-2) + \\ p_2 a(i - \Delta i_{k0}, j - \Delta j_{k0}) + p_2 a(i,j) \end{bmatrix},$$
$$3 \le l \le 1 + n - m,\ 1 \le i \le I,\ 1 \le j \le J.$$

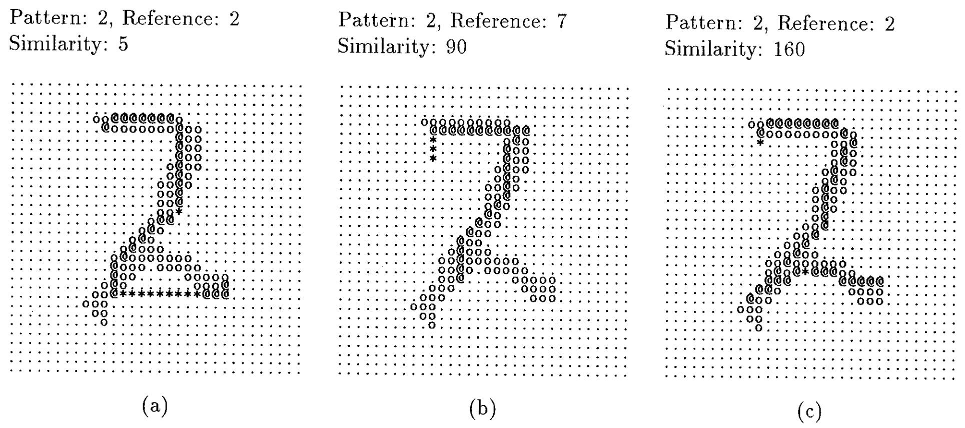
4. $g_g(i,j,l) =$
$$\max \begin{bmatrix} g_g(i - \Delta i_{k0}, j - \Delta j_{k0}, l-1) + p_0 a(i,j) \\ g_g(i - \Delta i_{k0} - \Delta i_{k1}, j - \Delta j_{k0} - \Delta j_{k1}, l-2) + \\ p_1 a(i - \Delta i_{k0}, j - \Delta j_{k0}) + p_1 a(i,j) \\ g_g(i - \Delta i_{k0} - \Delta i_{k2}, j - \Delta j_{k0} - \Delta j_{k2}, l-2) + \\ p_2 a(i - \Delta i_{k0}, j - \Delta j_{k0}) + p_2 a(i,j) \end{bmatrix},$$
$$2 + n - m \le l \le n,\ 1 \le i \le I,\ 1 \le j \le J.$$

5. $g(i,j,k) = g_g(i,j,n),$
$$1 \le i \le I,\ 1 \le j \le J. \qquad (20)$$

Regions will be assigned to each reference pattern stroke.

4 Experiment and Results

The objective of the experiment was to compare the recognition performance of the discussed matching algorithms and thus to get an idea about their relative merits and demerits and their behavior under different control parameters (priority coefficients).



Legend '.': White, 'o': Input Pattern, '*': Reference Pattern, '@': Both Input and Reference Patterns.

Fig. 6: Example of character '5' mis-recognized as character '6' in IRS but correctly recognized in 20° FRS. (a) & (b) Skeleton patterns corresponding to the correct and mis-recognized reference patterns. (c) Skeleton pattern for correct recognition in 20° FRS.

The character data used was the numeric character patterns from the ETL-6 database containing 1382 patterns for each of the numeric characters 0 to 9. As such 1382 × 10 = 13820 character patterns were used.

After extraction of the 64 × 63 size patterns, they were downsized to 32 × 32 size using a 2 × 2 filter. Noise cleaning was done by human assisted semi-automatic programs. During recognition, as preprocessing, the patterns were magnified/reduced and centered to coincide with a predefined rectangular frame in the pattern plane. The white meshes were also replaced by negative values to discourage any productive correspondence with them.

Reference patterns were manually optimized for each of the IRS, 45° FRS, 20° FRS and thus different standard patterns were used for each of the matching methods and priority coefficient values, in each case in order to maximize the individual average recognition percentage.

The experiment was done in a Sun Sparc Station.

The recognition accuracy for the three algorithms for various priority values are shown in Table 1. Example of a character pattern which was incorrectly recognized in inflexible IRS due to insufficient alignment with the reference pattern but was correctly recognized by 20° FRS is shown in Fig. 6.

5 Discussions

From the experimental results of Table 1, we observe that the flexible matching methods give significantly better performance than the inflexible one. A looser length boundary of reference pattern strokes may be used in inflexible IRS to better accommodate the input patterns. But as was observed by the authors in preliminary experiments, these too generous reference patterns while elevated the recognition percentage of certain characters, this also caused mis-recognition in other characters, and thus degrading the recognition performance in the long run.

In both the flexible matching methods, moderately distributed priority coefficients of (3, 2, 2) gave the highest average of correct recognition percentage. Though in both the cases (excepting (1, 1, 1) in 45° FRS) the variation in recognition percentage is small, a little investigation have shown that when both upright and slanted characters are present in the character database, or when characters are thin lined, an even distribution of (1, 1, 1) or a moderate distribution of (3, 2, 2) is a good choice, when only upright characters are present or characters are thick lined, a main vector enhanced priority coefficient distribution of (2, 1, 1) is to be used. 45° FRS with even priority distribution is a special case. Here the reference pattern vectors appear to be considerably unrestricted in direction, and thus making the recognition accuracy rate lowest of all the cases considered.

A problem observed during the course of experiment was that since the matching methods do not take into account the direction of input pattern strokes while carrying out correspondence, sometimes a short stroke matched a thick line segment of the input pattern at right angle with the direction of the input pattern stroke.

With respect to computational complexity and recognition time, definitely the IRS algorithm has simpler computational equations and lower complexity compared to the flexible algorithms. But in actual experiment, IRS needs greater number of strokes for satisfactory representation of the same reference patterns. Thus, average recognition time including skeletization for IRS (approx. 1.4 s) becomes comparable to those of the flexible ones (approx. 2.0 s and 2.1 s for 20° and 45° FRS respectively). And while the 20° and 45° FRS matching algorithms have the same order of computational complexity, the coordinate addressing involved in the former is more complicated than that of the latter. But in computer programs, by intelligent precalculation and storing of some group of terms, it is possible to make recognition time for the two almost entirely the same, as mentioned before.

6 Conclusions

We have investigated the optimization of one dimensional pattern to two dimensional DP–matching algorithm for hand–written character recognition. The algorithm analyzes dot matrix input, using the knowledge (reference pattern) represented in the form of directional vector sequence yielding the similarity measure and skeleton. Techniques of vector direction within a prespecified allowance have been newly proposed, improving the flexibility of matching. Recognition experiments based on the similarity measure were conducted, which showed the effectiveness of the proposed technique. The best flexibility allowance observed was approximately ±20° from the specified direction. How to use the skeleton (another output of matching) will be the next step of this study.

References

[1] Hiroaki Sakoe, "Recognition of Hand Written Characters Using Dynamic Programming (in Japanese)," *Proc. IECE Technical Meeting on Pattern Recognition and Learning*, PRL.74–20, Sept. 1974.

[2] Hiroaki Sakoe, "Recognition of Hand Written Characters Using Dynamic Programming (in Japanese)," *Electronics Magazine (Ohmu–sha)*, vol. 20, No. 8, pp. 850–854, Aug. 1975.

[3] Richard E. Bellman and Stuart E. Dreyfus, *Applied Dynamic Programming*. Princeton, NJ: Princeton University Press, 1962.

[4] Vladimir A. Kovalevsky, "Sequential Optimazation in Pattern Recognition and Pattern Description," *Information Processing 68*, pp. 1603–1607, North–Holland Publishing Company – Amsterdam, 1969.

[5] Hiroaki Sakoe and Seibi Chiba, "Dynamic Programming Algorithm Optimization for Spoken Word Recognition," *IEEE Trans. Acoust. Speech, Signal Processing*, vol. ASSP–26, No. 1, pp. 43–49, Feb. 1978.