

## **Notice of Violation of IEEE Publication Principles**

### **“Modified Syntactic Method to Recognize Bengali Handwritten Characters”**

by M.A. Rahman and A. El Saddik

in the IEEE Transactions on Instrumentation and Measurement, Vol 56, No. 6, December 2007, pp. 2623-2632

After careful and considered review of the content and authorship of this paper by a duly constituted expert committee, this paper has been found to be in violation of IEEE's Publication Principles.

This paper includes substantial copied portions of the work cited below, without attribution (including appropriate references to the original author(s) and/or paper title) and permission. The lead author, M.A. Rahman, was found to be solely responsible for this violation.

Due to the nature of this violation, reasonable effort should be made to remove all past references to this paper, and future references to this paper should also include a reference to the following article:

### **“Bengali Handwritten Character Recognition Using Modified Syntactic Method”**

by M.B. Islam, M.M.B. Azadi, M.A. Rahman, and M.M.A. Hashem

in the Proceedings of 2nd National Conference on Computer Processing of Bangla, 2005, pp. 264-275

# Modified Syntactic Method to Recognize Bengali Handwritten Characters

Md. Abdur Rahman, *Student Member, IEEE*, and Abdulmotaleb El Saddik, *Senior Member, IEEE*

**Abstract**—Several approaches have been proposed to recognize handwritten Bengali characters using different curve-fitting algorithms. The syntactic methods that are used to recognize printed Bengali alphabetic characters tend not to be suitable enough to properly distinguish Bengali handwritten characters. The proposed curve-fitting algorithm helps accurately recognize various strokes of different patterns (e.g., line and quadratic curve), thus drastically reducing the error elimination burden. As compared to existing approaches, the proposed modified syntactic method demonstrates significant improvement in recognizing Bengali handwritten characters.

**Index Terms**—Curvature, curve fitting, pattern recognition, syntactic method, thinning algorithm.

## I. INTRODUCTION

AFTER several decades of research activities, handwritten character recognition (HCR) still retains its appeal as an exciting and growing field of scientific investigation. HCR is the reading of a scanned image of a handwritten character along with an associated lexicon. The domain of handwritten character verification spans a variety of aspects. There are but a few applications, such as reading handwritten addresses in posted mail and automatically extracting information from diverse filled forms.

Depending on the language of which characters need to be recognized, different methods and techniques may be employed. An HCR system can be classified into two broad categories: 1) online and 2) offline. In online systems, handwriting is tracked, captured, and recognized during the writing process [1], [2]. In the case of offline recognition processes, recognition is performed on a captured static image once the writing is done [3]. Offline character recognition is subject to the attention of many researchers [4]–[7]. Much work has been done to recognize numeric characters [8]–[12]. However, alphabetic HCR poses more complexity than that of numeric characters. Few researchers have addressed alphabetic character recognition due to its inherent complexity; hence, HCR is still an open issue in this domain of research [5].

The syntactic method, as described in [13]–[17], is a well-developed approach to recognizing the characters of different Latin-based languages. One of the major advantages of the

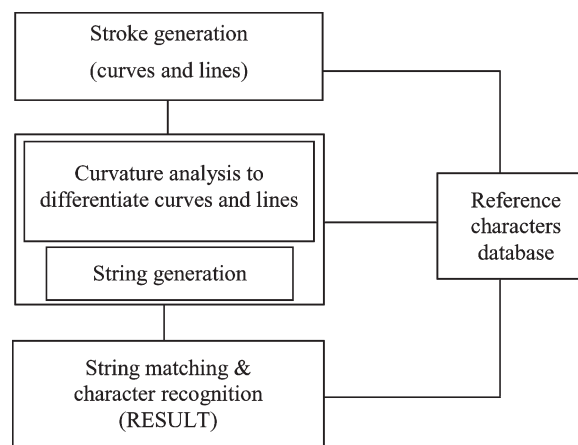


Fig. 1. System organization.

syntactic method is that it does not require any segmentation of the image whatsoever. Since segmentation is a prime source of error in HCR systems, this feature alone is important enough to support further adoption of this method. This method, however, is predictably lacking when it comes to recognizing non-Latin handwritten characters, particularly Bengali characters. The main difficulty with the recognition of Bengali handwritten characters is the vast variation in handwriting. Many factors account for the diversity in Bengali handwriting styles; among others are the regional origin of the writers, their educational level, and their profession. In fact, different circumstances such as stress, fatigue, and joy affect the handwriting process of any individual. In this paper, a new algorithm to identify various strokes of Bengali handwritten characters is proposed. It is a curve-fitting-based algorithm, which helps recognize various strokes of different patterns (e.g., line and quadratic curve) with high accuracy, as can be seen in the achieved results.

Fig. 1 depicts a high-level overview of our proposed modified syntactic method (MSM) system, which consists of four main components. In the reference characters database component, individual characters, which are provided by different authors, are scanned, preprocessed, and then stored in the reference characters database in a  $48 \times 48$  matrix format. In the stroke generation component, the reference character is broken into its primitive strokes such as curves and straight lines. The curvature analysis and string generation component implements our proposed curvature analysis algorithm to differentiate between curves and lines and generates the appropriate string consisting of the numeric codes of each stroke that uniquely identifies a character from its structural representation. Finally, the string

Manuscript received July 10, 2005; revised April 11, 2007. This work was supported by the Natural Sciences and Engineering Research Council of Canada.

The authors are with the School of Information Technology and Engineering, University of Ottawa, Ottawa, ON K1N 6N5, Canada.

Digital Object Identifier 10.1109/TIM.2007.907955

matching and character recognition component applies the dynamic programming paradigm to recognize the character.

The rest of this paper is organized as follows: Some closely related works are presented in Section II. An overview of the syntactic method is given in Section III. The proposed recognition process is explained in Section IV. Section V describes the experimental results. Finally, Section VI concludes this paper.

## II. RELATED WORK

Most researchers in the field of character recognition follow the decision theoretic approach [13], [14], which is generally only for classification of patterns and ignores structural information. The syntactic approach [13]–[17] has been chosen for this paper since this method not only classifies characters according to given patterns but also gives a structural description of them. The structural information of a character is important if we want to recognize a character, irrespective of its size.

Moshad and Ali [8] proposed an intelligent regional search method to recognize Bengali handwritten digits, which tend not to be suitable for Bengali HCR.

Rashid and Ali [9] argued about the string matching technique with dynamic programming. They used the eight-directional chain code method to generate primitives. Their test was only concerned with Bengali numerals. However, our technique is for the Bengali alphabet.

Pal and Chaudhuri have worked on unconstrained offline Bengali handwritten numerals [10], in which they have proposed a thinning and normalized free automatic recognition process. Still, the recognition rate is not as high as compared to our method.

The work described in [15] employs the visual pattern recognition system in which the authors recognize binary patterns from some distinct graphical symbols. The basic pattern recognition steps resemble our approach, except that their pattern recognition system uses predefined graphic symbols that are necessary to recover the visual position of an automatically guided vehicle, whereas, in our case, we recognize handwritten characters.

Several works have been done in the area of pattern and character recognition. Mia [18] selectively surveyed the printed Bengali characters in his thesis. He constructed numerical codes from the relationship among the strokes representing the structure of the character. There were several groups of characters that showed the same code due to the structural similarity, which was cited as a limitation of his approach.

Sattar and Rahman [19] discussed different problems of the recognition of printed Bengali characters by applying a template matching method. Their approach, however, differs from our approach in the recognition process and in recognition rates.

Sattar [20], in his thesis, used a decision theoretic pattern recognition scheme for recognizing Bengali printed alphanumeric characters.

Ali and Hossain [21] discussed the recognition of handwritten Bengali numerals by boundary line matching. The different

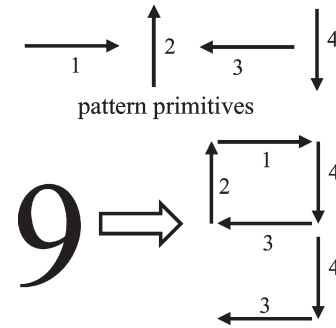


Fig. 2. Example of the digit “9” pattern primitives.

drawbacks of boundary line matching techniques are discussed in detail in their papers.

Rahman and Sattar [22] presented a detection mechanism based on some characteristic features. They divided the Bengali alphabet into four groups according to features: 1) upper part; 2) disjoint section; 3) vertical line; and 4) double vertical line.

Rasul [23], in his thesis, discussed Bengali character recognition using a genetic algorithm. He also used the concept of an artificial neural network. He clearly discussed the crossover and the mutation techniques on the recognition of Bengali characters. However, the recognition rate was not satisfactory.

Our research differs from the aforementioned papers by the technique that we followed to recognize Bengali handwritten characters, which is complemented by receiving excellent results.

## III. SYNTACTIC METHOD

Most of the pattern recognition research efforts of Gonzalez and Woods [14], Mia [18], and Marchand-Maillet and Sharaiha [24] during the last decade have dealt with the decision theory approaches and applications [14]. In some pattern recognition research, the structural information that describes each pattern is taken into consideration, and the recognition process includes not only the capability of assigning the pattern to a particular class but also that of describing some aspects of the pattern, which render it ineligible to be assigned to another class. A typical example of this approach is the scene analysis or the image and picture recognition process [17], [25], [26]. In the scene analysis process, the patterns under consideration are usually quite complex, and the required number of features is often very large, which makes the idea of describing a complex pattern, in terms of composition of simple subpatterns, very attractive. Indeed, when the patterns are complex and the number of possible descriptions is very large, it becomes impractical to regard each description as defining a specific class. The syntactic approach of pattern recognition creates a capability for describing a large set of complex patterns by using small sets of simple pattern primitives and grammatical rules. In fact, the practical effectiveness of such an approach depends on the ability to recognize the simple pattern primitives and their relationships, as represented by the composition operations. An example of the digit “9” pattern primitive and the corresponding structure is shown in Fig. 2 [14], [17], [27].

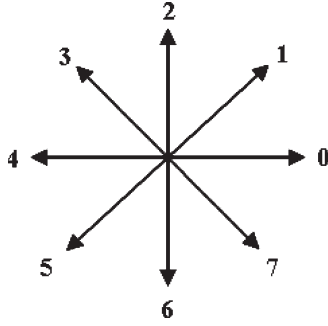


Fig. 3. Eight-directional chain code.



Fig. 4. Scanned Ka (ক).

The recognition of a primitive can also be accomplished based on a logical decision from the comparisons of features taken from the input component with a set of prespecified thresholds. As an example, straight lines can be classified by computing their lengths, slopes, or directions [18], [28]. The eight-directional chain codes [29] for straight lines are shown in Fig. 3.

#### IV. PATTERN RECOGNITION PROCESS

##### A. Input and Preprocessing

The scanned file of a handwritten character is either enhanced/compressed or restored to form a  $48 \times 48$  pixel size image. It is, however, not mandatory to produce  $48 \times 48$  dimension pixels. We decided to use this dimension because the utilized reference database is based on  $48 \times 48$  pixel characters. Hence, any dimension can be used. Once stored in the database, the process proceeds by alternately scanning from left to right and from top to bottom along various rows and columns of the input array. This continues until a black byte is encountered. A byte is considered as black if it has more than one black point. In Fig. 4, the scanned image of the letter Ka (ক) is shown.

To recognize a given character, first, the character needs to be digitized into a matrix of binary formats for ease of handling by the algorithm. The digitizer is a device that converts a physical sample to a pattern vector. It is often convenient, for recognition purposes, to arrange the pattern in the form of a pattern vector, i.e.,

$$\mathbf{X} = [X_1 \ X_2 \ X_3 \ \dots \ X_n]$$

where  $n$  is the number of measurements.

The component  $X_i$  of vector  $\mathbf{X}$  assumes the value of 1 or 0, depending on the state of the  $i$ th position for a particular input. Our algorithm operates on a representation of one character at

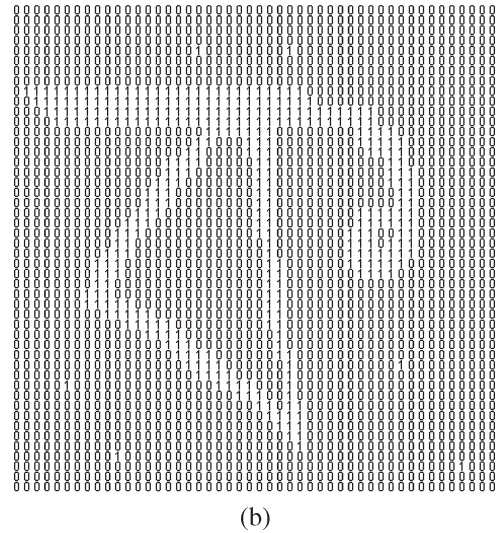
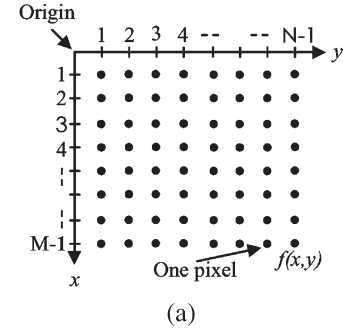


Fig. 5. (a) Digitization process. (b) Ka (ক) after digitization.

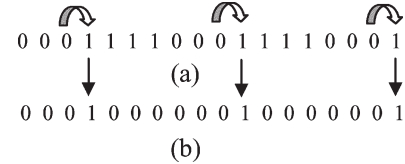


Fig. 6. (a) Before contour tracing. (b) After contour tracing.

a time. The representation is in the form of a matrix, whose entries have one of the two values of 0 or 1, corresponding to white and black points, respectively, in the original picture. The digitization process and digitized output of Ka (ক) are shown in Fig. 5(a) and (b), respectively.

##### B. Contour Tracing and Filtering

Contour tracing [15], [30] is the process of finding a series of black points on the boundary of a black region in a white field. The white-to-black transition points of the letter are obtained by scanning the digitized character each time from left to right and from top to bottom and storing the obtained values in a memory location.

The 0-to-1 transition and contour traced output are shown in Fig. 6(a) and (b). Fig. 7 illustrates the traced contour of the letter Ka (ক).

The next task toward pattern recognition is to filter out all the isolated 1s, called stray points, which are presented in the

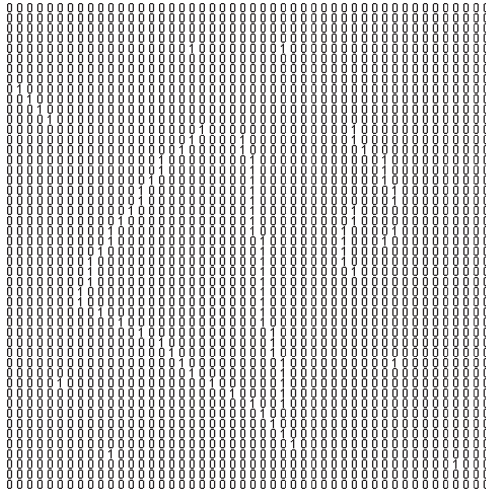


Fig. 7. Traced contour of Ka (ক).

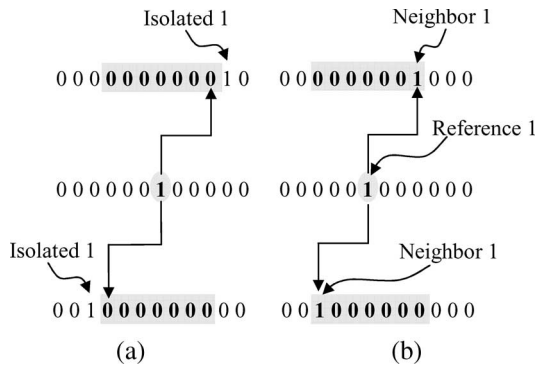


Fig. 8. (a) Isolated "1." (b) Digit "1" having neighbor points.

contour map of the character to be recognized. These stray points are those that are not associated with the alphabet; however, they arise during the scanning process and must be eliminated before the actual character recognition process starts. During the scanning process, a "1" is considered to be isolated if there is no similar "1" in its neighborhood region, defined as seven digits above and below the row of the reference digit, as shown in Fig. 8.

In Fig. 8, the middle row corresponds to the reference binary 1 under consideration. In Fig. 8(a), both digits "1" in the upper and lower rows are not neighbors of the reference digit "1" because they do not lie within the seven-digit region. In Fig. 8(b), both upper and lower row 1s are recognized as neighbors of the reference digit. Therefore, we keep them in the matrix. The process of filtering out unneeded digits is performed by applying the binary operator "AND" for each byte with the mask of its top and bottom bytes and then applying the "OR" binary operator on the computed results. The resulting value is stored in the matrix. For the first and last lines of the matrix, only the bottom and top bytes are taken, respectively, since the first line is neither bordered by a line above it, nor is the last line bordered by the one below it. After filtering out all stray points, the contour of Fig. 7 takes the form that is illustrated in Fig. 9.

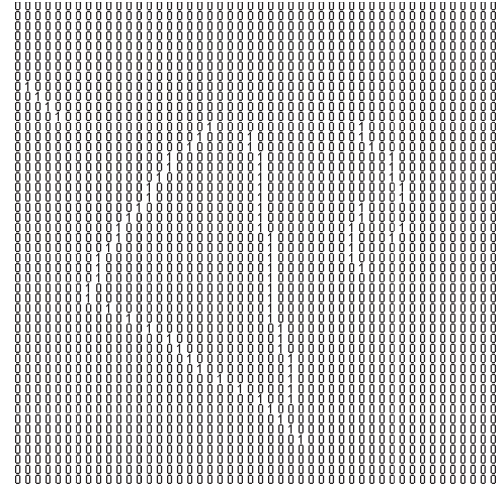


Fig. 9. After filtering Ka (ক).

### C. Strokes Extraction Process

To extract the strokes, all  $x$  and  $y$  coordinates of each pixel in the filtered file need to be saved [30]. As previously described, the scanning process starts from the top left corner and proceeds toward the bottom to find continuous points. Points in a line are deemed continuous if three bits on either side in its immediate upper line are black, as depicted in Fig. 8. If a discontinuity occurs, scanning proceeds to the next line with the assumption that the void in the current line is due to improper scanning.

If the next line presents continuity, a black point is assumed in the previous line because improper scanning may result in the loss of a pixel, and its coordinates are saved. Otherwise, the stroke is assumed to have terminated. All black points "1" of the stroke are terminated by a "0" during the storage process of the coordinates so that they are not reconsidered when the scanning process of searching for the next stroke starts again from the upper left corner [30]. The scanning strokes process continues until no new strokes are found. Fig. 10 shows the extraction of strokes from the filtered Ka (ক). The result is to produce the collection of stroke coordinates. We need this extraction process because we have to split up the image into several parts.

### D. Strokes Encoding Process

For the purpose of character recognition, it is desired to generate a numeric code for each character. The first step in this procedure is the selection of strokes and the generation of code for each stroke in a character.

1) *Selection of Strokes*: In the syntactic method, each input pattern is resolved into primitive structural elements, which are called strokes or morphs [30]. The first step in designing a syntactic model is the selection of appropriate morphs that can be represented in terms of patterns of interest. To simplify the selection process, we aim at spotting strokes that are simple to recognize and are minimum in number so that they are easy to find. For our purpose and analysis, six simple strokes were chosen to represent all 50 Bengali characters and ten numeric digits. These strokes are shown in Table I. The reason behind



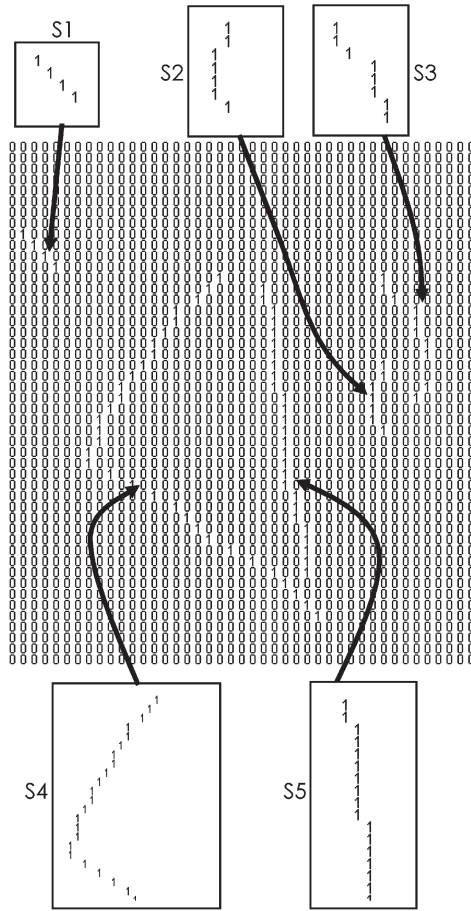


Fig. 10. Filtered strokes.

TABLE I  
STROKES USED FOR BENGALI CHARACTER RECOGNITION

Stroke	Characteristics	Numeric Code
—	Horizontal line	1
/	Positive sloping line	2
	Vertical Line	3
\	Negative sloping	4
∪	Vertical concave	5
∩	Vertical convex	6
None	Rejected stroke	0

choosing these strokes is obvious since all Bengali characters consist of lines or curves. Although there are characters that are a form of a circle, this is disregarded because when we side trace the letter, we only obtain the vertical convex or concave curves. Therefore, we take those lines or curves as a principle part of the character. As a result, a wide range of deviation is allowed in each of the six defined strokes, as presented in Fig. 11.

2) *Finding Stroke Code*: The coordinates of a particular stroke are now analyzed to find the appropriate code of that stroke. The strokes under consideration are either curves or straight lines with the main assumption that the characters of

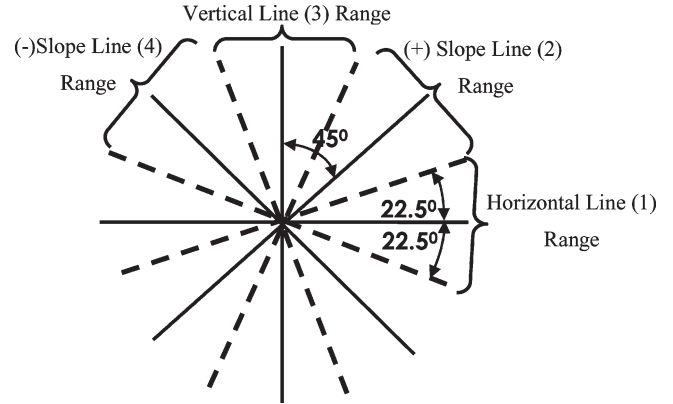


Fig. 11. Ranges of different straight lines.

any alphabet, in any language, can be represented by a combination of those strokes. Later, the coordinates of continuous strokes are assembled to find the appropriate numeric codes.

As shown in Fig. 10, among the five strokes of the alphabet Ka (ক), only S2, S3, and S4 should be curved, whereas S1 and S5 should be a straight line, from a theoretical point of view. By analyzing Fig. 10 with the naked eye, it can be guessed that S2 and S4 resemble numeric code 5 and that S3 takes the shape of numeric code 6 (see Table I). However, we intend to develop mathematical models to prove the fact. To form an equation from some points, we have several formulas. Among the equations, the following is very effective. If we set the points in the equation, then we get a path. The equation of curve fitting [31] is

$$x = a + by + cy^2 \quad (1)$$

where  $a$ ,  $b$ , and  $c$  are as follows:

$$a = \frac{\begin{vmatrix} \sum x & \sum y & \sum y^2 \\ \sum xy & \sum y^2 & \sum y^3 \\ \sum xy^2 & \sum y^3 & \sum y^4 \end{vmatrix}}{\begin{vmatrix} N & \sum y & \sum y^2 \\ \sum y & \sum y^2 & \sum y^3 \\ \sum y^2 & \sum y^3 & \sum y^4 \end{vmatrix}}$$

$$b = \frac{\begin{vmatrix} N & \sum x & \sum y^2 \\ \sum y & \sum xy & \sum y^3 \\ \sum y^2 & \sum xy^2 & \sum y^4 \end{vmatrix}}{\begin{vmatrix} N & \sum y & \sum y^2 \\ \sum y & \sum y^2 & \sum y^3 \\ \sum y^2 & \sum y^3 & \sum y^4 \end{vmatrix}}$$

$$c = \frac{\begin{vmatrix} N & \sum y & \sum x \\ \sum y & \sum y^2 & \sum xy \\ \sum y^2 & \sum y^3 & \sum xy^2 \end{vmatrix}}{\begin{vmatrix} N & \sum y & \sum y^2 \\ \sum y & \sum y^2 & \sum y^3 \\ \sum y^2 & \sum y^3 & \sum y^4 \end{vmatrix}}$$

where  $N$  is the total number of points on the stroke.

The point on the stroke that has the maximum curvature ( $\kappa$ ) [32] is calculated as follows:

$$\kappa = x_2 / (1 + x_1^2)^{3/2}. \quad (2)$$

Here,  $x_1 = dx/dy$ , and  $x_2 = d^2x/dy^2$ .

Taking the coordinates of 400 samples of filtered strokes from different scanned alphabets and then simulating (1) and (2) in MATLAB shows that if the value of  $\kappa \geq 0.9$ , then the stroke can be assumed as a curve; otherwise ( $\kappa < 0.9$ ), it is assumed that the stroke is a straight line. This new approach in differentiating curves and straight lines based on the value of  $\kappa$  will result in a significant minimization of errors and enhanced recognition, as we will demonstrate later.

Again, code 5 or 6 is assigned according to the value of a constant  $c$ , whether it is positive or negative. Fig. 10 illustrates the logical explanation of both codes 5 and 6. Three curves are represented in Fig. 10. Two of them are vertical concave because each of their slopes is positive, hence representing code “5.” The third one is considered negative due to the vertical convex curve, hence representing code “6.”

The slope of each line is calculated according to the following equation:

$$x = a + by. \quad (3)$$

Transforming (3) to  $y = mx + c$ , we get  $y = (x/b) - (a/b)$ . Thus, the slope of the line is equal to  $1/b$ , where  $a$  and  $b$  are defined as follows:

$$a = \frac{\begin{vmatrix} \sum x & \sum y \\ \sum xy & \sum y^2 \end{vmatrix}}{\begin{vmatrix} N & \sum y \\ \sum y & \sum y^2 \end{vmatrix}} \quad b = \frac{\begin{vmatrix} N & \sum x \\ \sum y & \sum xy \end{vmatrix}}{\begin{vmatrix} N & \sum y \\ \sum y & \sum y^2 \end{vmatrix}}.$$

Again,  $N$  is the number of points in a stroke of a given character. If  $N$  is less than or equal to 4, the stroke code is considered to be 0. This is explained by the fact that improper scanning can create up to four points. Therefore, setting that condition on the value of  $N$  can make up for this. A stroke with code “0” is created due to the existence of noise surrounding the character.

3) *Encoding of Characters*: This is the representation of characters in the form of a numeric string. The numerals in the code not only indicate strokes, which not only build the characters, but also show the relationship among the strokes. The strokes in the characters are traversed exactly one time from left to right and from top to bottom along various rows and columns. All strokes [30] are traversed exactly once. The letter code consists of the stroke codes written down in the order in which they were encountered. It is possible to get an idea about the structure of the character from its code. We started our scanning process from the top left corner and stopped at the bottom right corner.

### E. Recognition of Characters

1) *Matching Scores*: The characters represented by numeric string codes are now to be recognized. A search is conducted

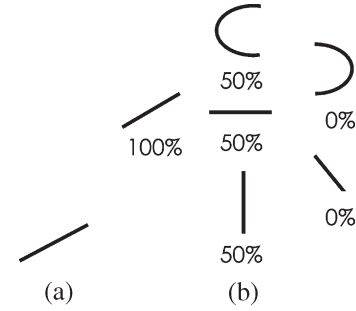


Fig. 12. (a) Reference segment. (b) Possible input segments and their matching scores.

Reference segments			
Input Segments	Rseg <sub>1</sub>	.....	Rseg <sub>k</sub>
Iseg <sub>1</sub>	S <sub>1,1</sub>		S <sub>1,k</sub>
Iseg <sub>2</sub>	S <sub>2,1</sub>		S <sub>2,k</sub>
.	.		.
.	.	.....	.
Iseg <sub>n</sub>	S <sub>n,1</sub>		S <sub>n,k</sub>

Fig. 13. Matching matrix.

on a dictionary of codes, called reference strings, to find a matching code corresponding to the code generated from the input character. The reference segment [9] and the matching scores [8], [9] for the corresponding input segments are shown in Fig. 12. As can be seen in Fig. 12(b), the leftmost straight line is identical to the reference segment; hence, it scores 100%. The curve and the straight line at the right top and bottom of the figure are completely opposite from those of the reference segment, thereby getting a 0% score. The rest of the input segments are scored in a similar process.

2) *Matching Matrix*: The reference pattern is thought of as a sequence of segments. To compare it with the input string, we need a reference string. We represent the reference string [9]  $R$  by


$$R = \{Rseg_1, Rseg_2, \dots, Rseg_k\} \quad (4)$$

whereas the input string  $I$ , which is also a list of segments, is represented by

$$I = \{Iseg_1, Iseg_2, \dots, Iseg_n\}. \quad (5)$$

Fig. 13 shows the matching matrix [9] that we have included to “weigh” the minimum score that the input deserves, where  $S_{i,j}$  denotes the matching score. Since we must compare it with previous data, the rows of the matrix are referenced by the input segments, and the columns are referenced by the reference segments. The intersection of the  $i$ th input segment and the  $j$ th reference segment holds the matching score between the  $i$ th and  $j$ th input segments.

TABLE II  
RELATIVE MATCHING SCORES FOR SPECIFIED PRIMITIVE CODES

		Reference Segment 					
		1	2	3	4	5	6
<div>Input Segment</div>	1	100	50	5	50	5	5
	2	50	100	50	0	70	0
	3	5	50	100	50	10	10
	4	50	0	50	100	0	70
	5	5	70	10	0	100	0
	6	5	0	10	70	0	100

Relative matching scores for specified primitive codes are shown in Table II. The numbers 1–6 in this table refer to the numeric code, as shown in Table I. Table II can be explained in the following way: when an input segment of numeric code type 1 is compared with a reference segment of numeric code type 1, the matching is perfect, which is 100%. Similarly, an input segment of code type 2 when matched with a reference segment of code type 6 produces a score of 0%. Other matching scores are mapped in a similar fashion.

3) *Optimal Score Computation*: After we have designed and built the matching matrix, we aimed at finding the optimal path through the matrix to achieve maximum pattern recognition scores. The search algorithm works as follows.

If we are currently in  $M(i, j)$ , we compute

$$Sl = M_{(i,j)} + \max((M_{(i+1,j)} + M_{(i+2,j)}), (M_{(i+1,j)} + M_{(i+2,j+1)})) \quad (6)$$

$$Sr = M_{(i,j)} + \max((M_{(i,j+1)} + M_{(i,j+2)}), (M_{(i,j+1)} + M_{(i+1,j+2)})) \quad (7)$$

$$Sd = M_{(i,j)} + M_{(i+1,j+1)} + \max(M_{(i+1,j+2)}, (M_{(i+2,j+2)} + M_{(i+2,j+1)})) \quad (8)$$

where  $Sl$  denotes a shift down,  $Sr$  denotes a shift right, and  $Sd$  denotes a shift diagonally.

Then, according to the maximum value that is found from  $\max(Sl, Sr, Sd)$ , we move down, right, or diagonally down into the matrix to the direction of the maximum value. Next, we clarify the directions obtained in Fig. 14 for Ka (ক) and Ba (ব) using equation (6)–(8).

For the case of Ka (ক), the following iterations produce the score values:

$$\begin{aligned} Sl &= 210 \quad Sr = 120 \quad Sd = 310 \\ \max(Sl, Sr, Sd) &= Sd, i = 1 \quad j = 1 \\ Sl &= 210 \quad Sr = 110 \quad Sd = 40 \\ \max(Sl, Sr, Sd) &= Sl, i = 1 \quad j = 2 \\ Sl &= 30 \quad Sr = 120 \quad Sd = 220 \\ \max(Sl, Sr, Sd) &= Sd, i = 2 \quad j = 1. \end{aligned}$$

Starting from  $M_{(1,1)}$ , which is 100, the aforementioned iterations result in the following sequences:  $Sd(M_{(2,2)}) \rightarrow$

$Sl(M_{(3,2)}) \rightarrow Sd(M_{(4,3)})$ , i.e., diagonal  $\rightarrow$  down  $\rightarrow$  diagonal. Averaging the obtained scores results in  $S_{av} = (M_{(1,1)} + M_{(2,2)} + M_{(3,2)} + M_{(4,3)})/4 = (100 + 100 + 100 + 100)/4 = 100\%$ .  $S_{av}$  for Ba (ব)  $= (M_{(1,1)} + M_{(1,2)} + M_{(2,3)} + M_{(3,3)} + M_{(4,3)})/5 = (0 + 100 + 100 + 100 + 10)/5 = 62\%$ . Other average matching scores are calculated in a similar fashion.

4) *Decision Taking*: After completing the optimal score computation, as previously described, we compute the average matching score of the input string with each reference string. An average matching score  $S_{av}$  of 90% or more is taken as a basis for a decision of recognition. Any outcome of  $S_{av}$  less than 90% is disregarded, and the input character is considered erroneous in this case and cannot be recognized with the presented work. Note that the reference string (character) that gives the maximum  $S_{av}$  with the input string is considered as the recognized character.

The extraction process produces a string that must be matched with predefined references. After performing the matching process with every reference, the matching percentage is calculated. For example, the average matching score computation of the input string 5336 with the reference strings 5365 (Ka (ক)), 453 (Ba (ব)), and 45 356 (Ra (র)) is shown in Fig. 14, where the optimal average score is 100% ( $> 90\%$ ), which matches with the reference string of the character Ka (ক). Therefore, the input character is Ka (ক).

## V. EXPERIMENTAL RESULTS

The test was performed on different Bengali characters written by various persons (adult). The system was implemented with Java on a Pentium-IV machine. The test subjects were told to naturally write each character, as if they were writing on a notebook. Table III shows the results of our test. Fifty samples for each of the characters, namely Ka (ক), Ba (ব), Ra (র), Bha (ভ), Jha (ঝ), Ri (ঝ), Da (দ), and Dha (ধ), were taken, making a total of 400 samples. Our pattern recognition system was able to recognize 380 characters correctly and 13 characters incorrectly and was unable to give a concrete decision in the remaining seven cases, where it rejected the input pattern. Accordingly, it can be concluded that our proposed algorithm gives 95% accuracy, on average, for every eight characters. Some ambiguity may occur, however, if a stroke in a character is small in comparison to the width and height of a given character. In other words, problems may occur if some of the strokes in a character are of the same magnitude as that of what is considered noise. The failures are due to improper scanning, stroke segments appearing at different angles, and strokes touching in one case but not in the other. Most of the problems occurred in the store strokes component. The store strokes routine would sometimes make mistakes if, for example, two strokes were very close together or one stroke was divided into two parts by more than three lines. Another source of failure may be due to unavoidable noise presented in the character.

From Table III, we can conclude that for some characters such as Ka (ক) and Ba (ব), the recognition rate is 98%, which is very high, whereas for other characters such as Jha (ঝ) and



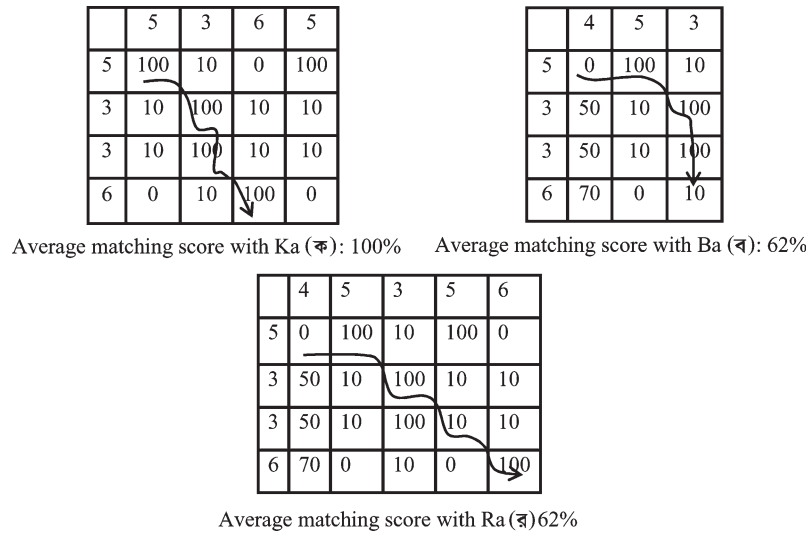


Fig. 14. Optimal matching score computation.

TABLE III  
RECOGNITION RATES OF DIFFERENT CHARACTERS

Character	No. of Samples taken (Handwritten)	No. of Samples Recognized	No. of Samples Wrongly Recognized	No. of Samples Rejected	% Accuracy
Ka (ক)	50	49	1	0	98
Ba (ব)	50	49	1	0	98
Ra (র)	50	48	1	1	96
Bha (ভ)	50	48	1	1	96
Jha (ঝ)	50	45	3	2	90
Ri (ঝ)	50	47	2	1	94
Da (দ)	50	48	1	1	96
Dha (ধ)	50	46	3	1	92

Dha (ধ), the recognition rate is 90% and 92%, respectively. After analyzing the scanned images that were provided by authors from diversified areas, it was found that the scanned images were very poorly written, although it could not be perceived by our own eyes. For clarification of the aforementioned fact, let us assume the case for Jha (ঝ). The authors wrote two vertical lines of Jha (ঝ) so closely that the software failed to separate those two lines and thereby recognized it as Ba (ব). By taking only standard scanned characters, the recognition rate of Jha (ঝ) was 98%.

In Table IV, we compare our proposed algorithm with some other algorithms [8]–[10]. Here, we applied our algorithm to the recognition of Bengali digits. From the table, we conclude that the recognition rate of Bengali digits by our proposed algorithm is better than that of other algorithms [8]–[10].

## VI. CONCLUSION AND FUTURE WORK

For the present analysis, the basic idea of the syntactic method was utilized to represent each complex pattern in terms of simple subpatterns. The techniques developed for resolving each character into a number of strokes [22], [24] and their

TABLE IV  
RECOGNITION RATES COMPARED TO OTHER RESEARCHES

Sample Character	Accuracy (in Percentage)			
	Proposed	Moshad[8]	Rashid[9]	Pal[10]
০ (Zero)	99	89.37	98	91.61
১ (One)	98	80	98	91.30
২ (Two)	97	82.5	96	91.02
৩ (Three)	98	88.75	96	89.78
৪ (Four)	95	93.12	62	94.92
৫ (Five)	93	87.5	69	89.72
৬ (Six)	95	88.75	90	88.52
৭ (Seven)	96	93.75	87	95.17
৮ (Eight)	97	93.75	85	97.06
৯ (Nine)	94	78.75	75	90.78

recognition lead to a new approach. However, the idea of the representation of characters by numeric codes for recognition purposes was proposed and successfully used in recognizing Latin letters, and we modified it so that it can now recognize Bengali characters.

The principal difficulty with this approach is the lack of use of a proximity criterion. If a character sample produces a code different from all known codes, it is rejected. As a partial solution to this problem, some of the characters have been

assigned several codes that correspond to common discrepancies between character samples due to a lack of uniformity in the quality of handwriting. Improvements in resolution would also be helpful.

The eventual target of designing a handwriting recognition system with 100% accuracy is illusionary because even human beings are not able to perfectly recognize handwritten text, e.g., it happens to most people that sometimes they cannot even read some of their own written characters. There will always be an obligation for the writer to write clearly. Humans roughly recognize 96% [33].

By considering the curve fitting with curvature [32], the performance can be surprisingly improved. Our method proved to be useful as it was successful in recognizing about 95% of the characters. With some modifications to the proposed algorithms, compound words, including both characters and numerals, can also be recognized, examples of which being postal code recognition, license plate recognition, etc.

Several possibilities exist that could improve the chances of constructing a practical text recognition device [14], such as a high standard of scanning quality, stylized character [27], [34], [35], size, and language simplification.

As possible future work, we would aim at improving the percentage of correctly recognized characters by dynamically assigning edit distances, weights of insertion, deletion, and substitution values and by using some training processes. Effort should also be given to make the algorithm scalable (multi-threaded) so that multiple characters can be read or a whole page can be recognized simultaneously. We would also like to perform an evaluation of the proposed method on nonnumeric alphabetic characters. Comparing the proposed method with similar works done to recognize other languages such as Chinese, Arabic, and French was out of the scope of this paper but may be addressed in future work.

The problem would be the development of a practical Bengali character recognition machine, toward which the effort of this project is directed. It is our hope that advances in this area will provide additional incentive for work in translation devices.

## REFERENCES

- [1] A. M. Namboodiri and A. K. Jain, "Online handwritten script recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 1, pp. 124–130, Jan. 2004.
- [2] C.-L. Liu, S. Jaeger, and M. Nakagawa, "Online recognition of Chinese characters: The state-of-the-art," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 2, pp. 198–213, Feb. 2004.
- [3] T. Steinherz, E. Rivlin, and N. Intrator, "Offline cursive script word recognition—A survey," *Int. J. Doc. Anal. Recognit.*, vol. 2, no. 2/3, pp. 90–110, Dec. 1999.
- [4] A. Vinciarelli, "A survey on off-line cursive word recognition," *Pattern Recognit.*, vol. 35, no. 7, pp. 1433–1446, Jul. 2002.
- [5] F. Bortolozzi, Jr., A. Britto, L. S. Oliveira, and M. Morita, "Recent advances in handwriting recognition," in *Proc. Int. Workshop Document Anal.*, 2005, pp. 1–30.
- [6] V. K. Govindan and A. P. Shivaprasad, "Character recognition—A review," *Pattern Recognit.*, vol. 23, no. 7, pp. 671–683, Jul. 1990.
- [7] A. L. Koerich, R. Sabourin, and C. Y. Suen, "Large vocabulary off-line handwriting recognition: A survey," *Pattern Anal. Appl.*, vol. 6, no. 2, pp. 97–121, Jun. 2003.
- [8] M. A. A. Moshad and M. M. Ali, "Recognition of handwritten Bangla digits by intelligent regional search method," in *Proc. Int. Conf. Comput. Inf. Tech.*, Dec. 2001, pp. 297–302.
- [9] A. M. Rashid and M. M. Ali, "On line recognition of handwritten characters using 1-dim, 1-dim DP approach," in *Proc. Int. Conf. Comput. Inf. Tech.*, Dec. 1998, pp. 86–90.
- [10] U. Pal and B. B. Chaudhuri, "Automatic recognition of unconstrained off-line Bangla handwritten numerals," in *Proc. Advances Multimodal Interfaces*, T. Tan, Y. Shi, and W. Gao, Eds., 2000, vol. 1948, pp. 371–378.
- [11] P. Gader, B. Forester, M. Ganzberger, A. Gillies, B. Mitchell, and T. Youcum, "Recognition of handwritten digits using template and model matching," *Pattern Recognit.*, vol. 24, no. 5, pp. 421–431, 1991.
- [12] H. Mitoma, S. Uchida, and H. Sakoe, "Online character recognition based on elastic matching and quadratic discrimination," in *Proc. 8th ICDAR*, Seoul, Korea, 2005, pp. 36–40.
- [13] N. Efford, *Digital Image Processing: A Practical Introduction Using Java*. Boston, MA: Addison-Wesley, 2000.
- [14] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*. New York: Addison-Wesley, 2002.
- [15] H. Khalfallah, E. M. Petriu, and F. C. A. Groen, "Visual position recovery for an automated guided vehicle," *IEEE Trans. Instrum. Meas.*, vol. 41, no. 6, pp. 906–910, Dec. 1992.
- [16] M. G. Thomason and R. C. Gonzalez, "Syntactic recognition of imperfectly specified patterns," *IEEE Trans. Comput.*, vol. C-24, no. 1, pp. 93–96, Jan. 1975.
- [17] E. Tanaka, "Theoretical aspects of syntactic pattern-recognition," *Pattern Recognit.*, vol. 28, no. 7, pp. 1053–1061, Jul. 1995.
- [18] M. A. K. Mia, "Recognition of printed Bangla characters by syntactic method of pattern recognition," M.S. Eng. thesis, Dept. Comput. Sci. Eng., Bangladesh Univ. Eng. Technol., Dhaka, Bangladesh, 1992.
- [19] M. A. Sattar and S. M. Rahman, "An experimental investigation on Bangla character recognition system," *Bangladesh Comput. Soc. J.*, vol. 4, pp. 1–4, Dec. 1989.
- [20] M. A. Sattar, "Recognition of printed Bangla characters by decision theoretic method pattern recognition," M.S. Eng. thesis, Dept. Comput. Sci. Eng., Bangladesh Univ. Eng. Technol., Dhaka, Bangladesh, 1990.
- [21] M. M. Ali and M. B. Hossain, "Recognition of handwritten Bangla numeral by boundary line matching," *J. CSI*, vol. 32, no. 2, pp. 13–18, Jun. 2002.
- [22] A. F. R. Rahman and M. A. Sattar, "An analysis of Bengali characters: Detection of some characteristic features," in *Proc. Int. Conf. Comput. Inf. Tech.*, Dec. 1998, pp. 142–145.
- [23] G. Rasul, "Bengali character recognition using genetic algorithm," M.S. Eng. thesis, Dept. Comput. Sci. Eng., Bangladesh Univ. Eng. Technol., Dhaka, Bangladesh, 1998.
- [24] S. Marchand-Maillet and Y. M. Sharaiha, *Binary Digital Image Processing: A Discrete Approach*. New York: Academic, 2000.
- [25] A. Rosenfeld and A. C. Kak, *Digital Picture Processing*, 2nd ed., vol. 1/2. New York: Academic, 1982.
- [26] N. Memon, D. L. Neuhoff, and S. Shende, "An analysis of some common scanning techniques for lossless image coding," *IEEE Trans. Image Process.*, vol. 9, no. 11, pp. 1837–1848, Nov. 2000.
- [27] M. T. Hoque and M. M. Ali, "Noise removal algorithm from images by using breadth first search strategy," in *Proc. Nat. Conf. Comput. Info. Syst.*, Dec. 1997, pp. 156–160.
- [28] J. Park, "An adaptive approach to offline handwritten word recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 920–931, Jul. 2002.
- [29] U. Garain and B. B. Chaudhuri, "Recognition of online handwritten mathematical expressions," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 6, pp. 2366–2376, Dec. 2004.
- [30] T. W. Sze and Y. H. Yang, "A simple contour matching algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-3, no. 6, pp. 676–678, Nov. 1981.
- [31] S. C. Chapra and R. P. Canale, *Numerical Methods for Engineers: With Software and Programming Applications*, 4th ed. India: Tata McGraw-Hill, 2002.
- [32] L. Euler, *Foundations of the Differential Calculus*. New York: Springer-Verlag, 2000. J. D. Blanton (translator).
- [33] S. Haigh, *Network Notes #7*, 1996, International Services. National Library of Canada.
- [34] A. Bishnu and B. B. Chaudhuri, "Segmentation of Bangla handwritten text into characters by recursive contour following," in *Proc. Int. Conf. Doc. Anal. Recogn.*, Sep. 1999, pp. 402–405.
- [35] M. C. Fairhurst and A. F. R. Rahman, "Generalised approach to the recognition of structurally similar handwritten characters using multiple expert classifiers," *Proc. Inst. Electr. Eng.—Vis. Image Signal Process.*, vol. 144, no. 1, pp. 15–22, Feb. 1997.



**Md. Abdur Rahman** (S'04) received the B.Sc. degree in electrical and electronic engineering from Bangladesh University of Engineering and Technology, Dhaka, Bangladesh, in 2000 and the M.Sc. degree in electrical engineering from the University of Ottawa, Ottawa, ON, Canada, in 2005. He is currently working toward the Ph.D. degree in electrical engineering with the School of Information Technology and Engineering, University of Ottawa.

In 2001, he joined the Department of Computer Science and Engineering, Khulna University of Engineering and Technology, Khulna, Bangladesh, as a Lecturer. His research areas are in multimedia communication, smart sensors, telesurveillance, and distributed networks.



**Abdulmotaleb El Saddik** (M'02–SM'03) received the Dipl.-Ing. and Dr.-Ing. degrees from Darmstadt University of Technology, Darmstadt, Germany, in 1995 and 2001, respectively.

He is the University Research Chair and an Associate Professor with the School of Information Technology and Engineering (SITE), University of Ottawa, Ottawa, ON, Canada, the Director of the Multimedia Communications Research Laboratory, SITE, and the Director of the Information and Communications Technology Cluster, Ontario Research

Network on E-commerce. He is an Editor of the International Journal of Advanced Media and Communication and an Associate Editor of the ACM Journal of Educational Resources in Computing. He is the author or coauthor of three books and more than 150 publications. He is a leading researcher in collaborative haptic audiovisual environments, service-oriented architectures, and ambient interactive media and communications.

Prof. El Saddik is a Distinguished IEEE Lecturer and has served as a Guest Editor for several IEEE transactions and journals. He was a recipient of the Premier's Research Excellence Awards in 2004 and the Alexander von Humboldt-Friedrich Wilhelm Bessel Research Award in 2007.