Writer-independent Offline Recognition of Handwritten Ethiopic Characters

Yaregal Assabie and Josef Bigun
School of Information Science, Computer and Electrical Engineering
Halmstad University, Halmstad, Sweden
{Yaregal.Assabie, Josef.Bigun}@hh.se

Abstract

This paper presents writer-independent offline handwritten character recognition for Ethiopic script. The recognition is based on the characteristics of primitive strokes that make up characters. The spatial relationships of primitives whose combinations form complex structures of Ethiopic characters are used as a basis for recognition. Although this approach efficiently recognizes properly written characters, the recognition rate drops for characters where the spatial relationships of their primitives could not be drawn. This happens mostly when the connections between primitives are not properly written, which is a common case in handwriting. To complement the recognition, we classify characters based on the characteristics of their primitives, resulting in grouping of characters in a five-dimensional space. Once the type of characters is identified, recognition can be achieved with a minimal set of information from their spatial relationships. A comprehensive database is also developed to standardize the evaluation of research works on offline Ethiopic handwriting recognition systems. Our proposed system is tested is with the database and experimental results are reported.

Keywords: Ethiopic, Handwriting Recognition, Database.

1. Introduction

Recognition of offline handwritten documents is one of the extensively studied pattern recognition problems. Due to the complexity of the problem, however, it is still a subject of research for the pattern recognition community. Several approaches have been proposed for recognition of handwritten characters. The most commonly used techniques are neural networks, Hidden Markov Model (HMM), elastic matching, stroke analysis, and a combination of multiple classifiers [3], [6], [8]. The recognition rate produced by each technique varies depending on, among other things, the nature of the script. The techniques have been applied widely on writing systems such as Latin, Chinese, Japanese, Arabic, and Indian. Nevertheless, only little has been studied on handwriting recognition for Ethiopic script in general.

Ethiopic script is used effectively over the past two millennia as a writing system for languages spoken in Ethiopia, currently with a population of over 80 million. The script has been largely used by Geez and Amharic, which are the liturgical and official languages of the country, respectively. Although Ethiopic alphabet has recently been standardized to have 435 characters, roughly half of them are used practically in daily communications by Amharic and other major languages. Ethiopic is a modification-based script where the modifiers (‘vowels’) are usually added to the base character to give a derived vocal sound. Sometimes, the modification can also be achieved by slightly deforming the shape of the base character. The alphabet is conveniently written in a tabular format of seven columns (orders) where the first column represents the base character and other columns represent derived vocal sound of the base character. In most cases, the modification process for each order forms a pattern.

2. Database Development

Databases are important to train, test, and compare character recognition systems. However, there is no database of offline handwritten documents developed so far for Ethiopic script. Thus, we developed a database to standardize the evaluation of research works on recognition of handwritten Ethiopic script. The documents are scanned in grayscale format with a resolution of 300dpi. They are collected from two sources and divided as *Group I* (Ethiopian Orthodox Church documents) and *Group II* (documents from ordinary writers). Each group of the database is divided into test (70%) and training (30%) sets.

Most of *Group I* documents are prepared many years back. Until the introduction of modern printing machines, both liturgical and state documents had been prepared by the Church. The writing style used by the Church is different from ordinary writing style in that church documents are written carefully by ink, and characters are not cursive and connected as well. Since most of them are age-old documents, they are degraded and noisy as compared to documents written recently on white papers. From *Group I* documents, a total of 114 pages written in Geez and Amharic languages are included in the database. The inclusion of such documents in the database is also useful for analysis and recognition of historical documents.

Group II is collected from various native users of the script divided into two subgroups as *Group IIa* and *Group IIb*. In *Group IIa*, 59 page document dealing with various issues and news such as education, sport, politics, economics, and other social events is prepared. Each page is given to three different people to write the page content by hand on a white paper using a pen of their own. Thus, a total of about 379,800 character samples from 177 writers are included in this subgroup. Since the document is prepared randomly from real-world texts, character samples are not evenly distributed. *Group IIa* can be used to evaluate line detection and character/word segmentation algorithms.

Group IIb is collected from another group of 152 participants writing each character in the Ethiopic alphabet three times. Although there are currently 435 characters in the Ethiopic alphabet, many of them are designed recently for minority languages and they are not used practically in daily communications by the official language Amharic and other major languages. Thus, the subgroup includes the 34 base characters and their derivatives (i.e., 34x7=238 characters), which are commonly used by most Ethiopian languages including Amharic. Despite their rare appearance in texts, the subgroup also includes the so called labialized characters listed as the following: ሏ ሟ ሯ ሷ ሿ ቋ ቧ ቷ ቿ ኋ ኗ ኟ ኳ ዟ ዧ ዷ ጇ ጓ ጧ ጯ ጿ ፏ ቈ ኰ ኈ ጐ and ኧ. Thus, a total of 265 Ethiopic character symbols are included in the subgroup. Therefore, *Group IIb* has 120,840 character samples where each sample is evenly distributed in the subgroup. This subgroup of the database can be used for evaluating recognition of isolated handwritten characters. Figure 1 shows sample images from the database.

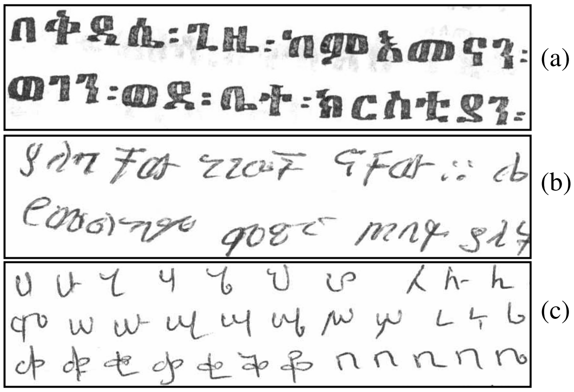


Figure 1. Sample images taken from (a) Group I, (b) Group IIa, and (c) Group IIb.

3. The Handwriting Recognition System

The proposed offline handwriting system combines the structural/syntactic properties of primitive strokes and the types of characters classified based on their salient primitives. The structural and syntactic model encodes the orientation, structure, relative length, and spatial position of primitive strokes which are extracted by making use of the direction field tensor. A special tree structure is used to handle the relationship and the tree is traversed to generate a set of unique sequence of primitive strokes for each character. The generated sequence of strokes is matched against a knowledge base which stores possibly occurring sequences of primitive strokes for each Ethiopic character. The characteristics of primitives are used to classify characters into a five-dimensional space and help recognize unknown input with little information from the spatial relationships of primitives. The flowchart of the recognition system is shown in Fig. 2 and the details are presented below.

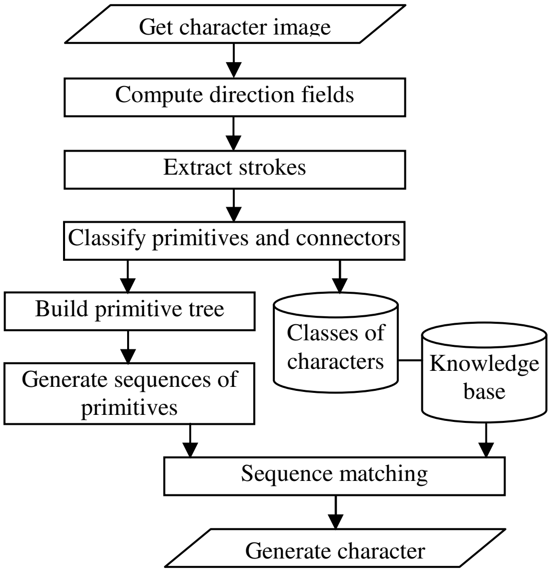


Figure 2. The offline handwriting recognition system.

3.1. Computing direction fields

Direction field tensor $\mathbf{S}$ is a 2x2 matrix which computes the optimal direction of pixels in a local neighborhood of an image f. The tensor is introduced further in detail in [2] and it is computed as:

$$S = \begin{pmatrix} \iint (D_x f)^2 dxdy & \iint (D_x f)(D_y f)dxdy \\ \iint (D_x f)(D_y f)dxdy & \iint (D_y f)^2 dxdy \end{pmatrix} \qquad \mathbf{(1)}$$

The integrals are implemented as convolutions with a Gaussian kernel, and D_x and D_y are derivative operators. The local direction vector is the most significant eigenvector modulated by the error differences (the difference of eigenvalues). This vector field is also known as the *linear symmetry* (LS) vector field and can be obtained directly by use of complex moments. The latter are defined as:

$$I_{mn} = \iint ((D_x + iD_y)f)^m ((D_x - iD_y)f)^n \, dxdy \qquad (2)$$

where m and n are non-negative integers. Among other orders, of interest to us are I_{10}, I_{11}, and I_{20} derived as:

$$I_{10} = \iint ((D_x + iD_y)f)dxdy \qquad (3)$$

$$I_{11} = \iint \left|(D_x + iD_y)f\right|^2 dxdy \qquad (4)$$

$$I_{20} = \iint ((D_x + iD_y)f)^2 dxdy \qquad (5)$$

In a local neighborhood of an image, I_{10} computes the ordinary gradient field; I_{11} measures gray value changes (the sum of eigenvalues of $\boldsymbol{S}$); and I_{20} gives a complex value where its argument is the optimal direction of pixels in double angle representation and its magnitude is the local LS strength (the difference of eigenvalues of $\boldsymbol{S}$). Pixels with low magnitude are said to be lacking LS property. As shown in Fig. 3, I_{10} and I_{20} images can be displayed in color where the hue represents direction of pixels with the red color corresponding to the direction of zero degree.

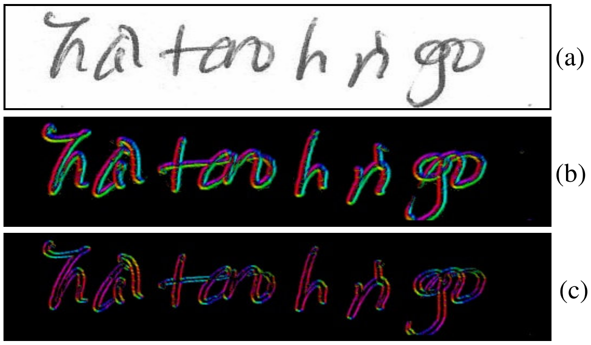


Figure 3. (a) Handwritten text, (b) I_{10} of ***a***, (c) I_{20} of ***a***.

3.2. Primitive strokes

For machine printed text, Ethiopic characters can be effectively represented only by *seven* types of vertically and diagonally oriented primitive structures listed, with example characters in brackets, as: *long vertical line* (ዘ), *medium vertical line*(ስ), *short vertical line* (ሐ), *long forward slash* (ጊ), *medium forward slash* (ፊ), *backslash* (ለ), and *appendages* (ቺ). The horizontal lines are considered as connectors of primitive structures. The recognition system for machine printed Ethiopic characters is further described in detail in [1]. However, in the case of handwritten text, characters are written with various shapes since writers are not perfectly writing like printed ones. Therefore, we redefined primitives for effective representation of handwritten characters. Primitive strokes for handwritten characters are hierarchically classified based on their orientation/structure type, relative length with in the character, and relative spatial position. For the purpose of computation, each classification level is assigned with numbers ranging from 6 to 9. The hierarchy of classification is given as follows.

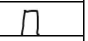
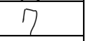
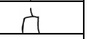
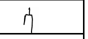
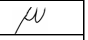
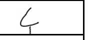
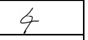
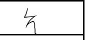
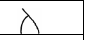
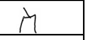
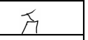
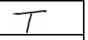
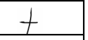
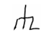
i. **Orientation/structure type**: There are three groups of orientations for primitive strokes namely, *forward slash* (9), *vertical* (8), and *backslash* (7). *Appendages* (6) do not fit to a specific orientation. Rather, they are recognized by their structure type in the case of machine printed text, e.g. in ቲ. In handwritten text, *appendages* are usually not marked well and we define them as the end points of horizontal lines as in ተ.

ii. **Relative length**: The orientation of primitives is further classified based on their relative length as *long* (9), *medium* (8), and *short* (7). Long is defined as a primitive that runs from the top to the bottom of the character, where as short is a primitive that touches neither the top nor the bottom of the character. *Medium* refers to a primitive that touches either the top or the bottom (but not both) of the character. Due to their small size, *appendages* are always considered as *short*.

iii. **Relative spatial position**: At this level of classification hierarchy, primitives are further classified according to their spatial position with in the character as *top* (9), *top-to-bottom* (8), *bottom* (7), and *middle* (6). *Short* primitives can only have a relative spatial position of *middle*. *Top-to-bottom* position applies to *long* primitives which run from the top to the bottom of the character. Primitives with *medium* relative size can have either *top* or *bottom* spatial position. *Appendages* may appear at the *top*, *middle*, or *bottom* of the character.

The above classification scheme results in 15 types of primitive strokes, which are used to represent all the 435 Ethiopic characters. Table 1 summarizes the list of primitive strokes and their numerical codes.

Table 1. Hierarchical classification of primitives.

Orientation/ Structure	Length	Position	Code	Example Character
Vertical	Long	Top-to-bottom	898	
	Medium	Top	889	
		Bottom	887	
	Short	Middle	876	
Forward Slash	Long	Top-to-bottom	998	
	Medium	Top	989	
		Bottom	987	
	Short	Middle	976	
Backslash	Long	Top-to-bottom	798	
	Medium	Top	789	
		Bottom	787	
	Short	Middle	776	
Appendage	Short	Top	679	
		Middle	676	
		Bottom	677	

Primitives are extracted by making use of I10 and I20. The I20 image is used to group pixels into parts of primitives and connectors based on the direction information. After converting the double angle of I20 into a simple angle representation (by halving the argument of I20), pixels with LS properties and directions of [0..60] degrees are considered as parts of primitives and those with directions of (60..90] degrees are considered as parts of connectors. The extracted linear structures in the I20 image are mapped onto the I10 image to classify them into left and right edges of primitives. A primitive is then formed from the matching left and right edges. After primitives are extracted, they are further classified using their direction information, relative length, and spatial position.

3.3. Spatial relationships of primitives

Based on our structural and syntactic analysis, horizontal strokes in characters are considered as connectors of primitive strokes. The way primitives are connected to each other is referred as spatial relationship. A primitive can be connected to another at one or more of the following regions of the strokes: *top*, *middle*, and *bottom*. The first connection detected as one goes from top to bottom is considered as *principal connection*. The principal connection is used to determine the position of spatial relationships between two primitives. Other additional connections, if there exist, are *supplemental* connections. A total of 18 connection types are identified between primitives of the Ethiopic characters and a summary is given in Table 2. Connection regions between primitives are also assigned with numbers as: *top* (1), *middle* (2), and *bottom* (3). The number 4 is used for cases where there is no connection. A connection between two primitives is represented by ***xy*** where ***x*** and ***y*** are numbers representing connection regions for the *left* and *right* primitives, respectively.

The spatial relationships of primitives of a character is handled in a special tree structure known as *primitive tree*. Based on the interconnection analysis of primitives in Ethiopic characters, a general primitive tree structure is designed as shown in Fig. 4.

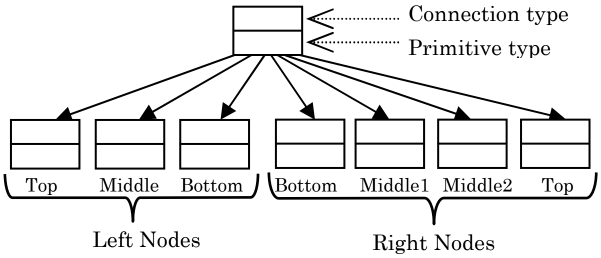


Figure 4. General structure of primitive tree.

Table 2. Connection types between primitives.

Principal connection	Number of supplementary connections					
	None	One			Two	
					22	22
		23	32	33	32	33
11						
	11	1123	1132	1133	112232	112233
12						
	12		1232			
13						
	13					
21						
	21	2123	2132	2133		
22						
	22					
23						
	23					
31						
	31					
32						
	32					
33						
	33					

The primitive tree inherits the properties of binary search trees by arranging the primitives as left and right. The two middle nodes in the right nodes is due to connection of two primitives at the middle of the right of their parent primitives for some characters like and . A primitive stroke spatially located at the left top position of the character is selected as a root node. Other primitives are built recursively into the tree in such a way that primitives connected to the left of a primitive are added to the left and those connected to the right are added to the right side based on the principal connection. The child nodes correspond to the possible number of primitives (three to the left and four to the right) connected to the parent primitive. Figure 5 shows an example of primitive tree for the character .

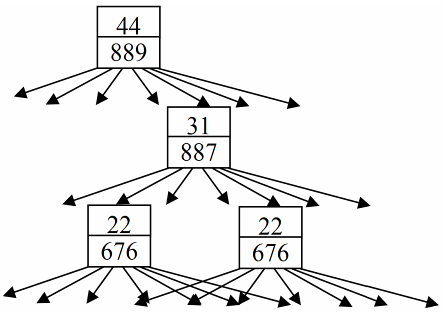
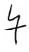
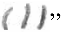
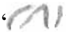


Figure 5. Primitive tree for the handwritten character .

3.4. Matching sequences

To reduce the computational cost of primitive trees, they are converted to string data structure by traversing them in the order of {**left**{top, middle, bottom}, **parent**, **right**{bottom, middle1, middle2, top}}. This traversal generates a unique sequence of primitives and their connections in the form of string data. For example, the primitive tree in Fig. 6 is converted to string data as: {44,889,22,676,31,887,22,676}. For each character in the Ethiopic alphabet, possibly occurring sequences of primitive strokes and their connections are stored as a knowledge base. Recognition of unknown input is achieved by matching their sequence of primitive strokes and connections against the knowledge base. During sequence matching, the similarity between the unknown input and each record in the knowledge base is computed and the best match is considered for recognition, which is decided based on a similarity threshold.

3.5. Classification of characters

The recognition process explained above works efficiently provided that spatial relationships between primitives are drawn. Although spatial relationships of primitives are fairly extracted for machine printed characters, it may not be a common case for handwritten characters. We observed that connections between primitives receive attention from writers when there are structurally similar characters so as to remove ambiguities between them. For example, it is difficult to recognize “” as there are ወ, ጠ, and ሠ which are formed from three long primitives but differ only in how they are connected. With this knowledge, writers tend to clearly write connectors between primitives of such characters. On the other hand, መ can be written as “” without the primitives connected properly at their top and bottom, but still easily recognized by native users since there is no other character formed only from four long primitives.

Thus, we classify characters based on their primitives, the purpose of which is to form groups of characters where each group has structurally similar characters. For classification of characters, we use a set of features from the primitives listed as: *number of salient primitives (s)*, *number of long primitives (l)*, *number of salient primitives touching the top of the character (t)*, *number of salient primitives touching the bottom of the character (b)*, and *number of appendages (a)*. In this case long, medium and short primitives, which have a defined relative length, are considered as salient primitives. Appendages are not included since they are just defined as end points of horizontal lines having no vertical length. Then, each character C is represented by a feature vector as $C = (s, l, t, b, a)$ where $s \in \{1, 2, …, 8\}$, $l \in \{0,1, 2, 3, 4\}$, $t \in \{1, 2, 3, 4\}$, $b \in \{1, 2, …, 6\}$, and $a \in \{1, 2, …, 6\}$. Then, based on their feature vector values, characters are grouped and stored in to cells of a five-dimensional space of size 8x5x4x6x6. Eventually, structurally similar characters are stored in the same cell or its neighborhood. One or more characters are placed in a cell, and sometimes cells may also contain no characters. Figure 6 shows part of the character classes with neighboring groups interconnected to each other.

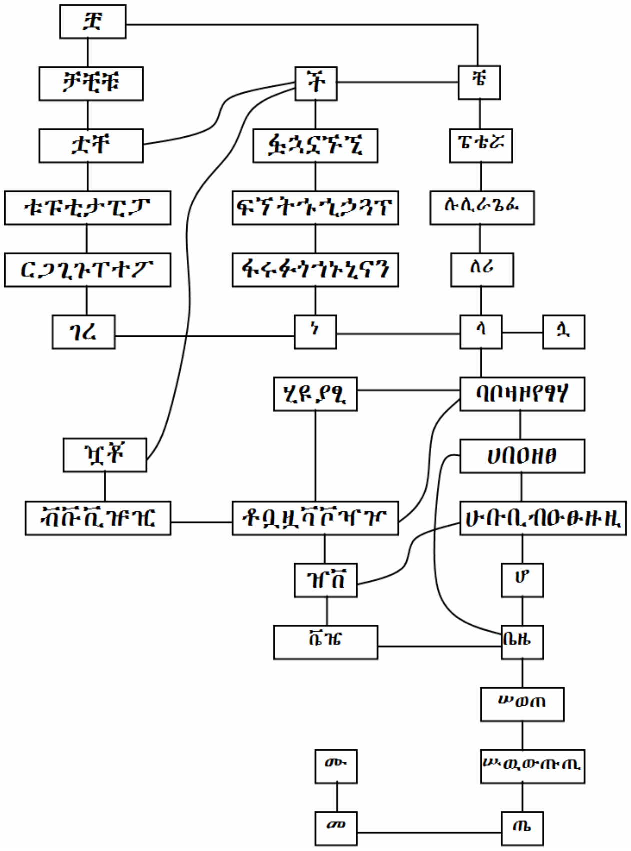


Figure 6. Part of the character classification with their neighborhoods linked to each other.

During the recognition process, the unknown input is classified using its feature vector to locate the cell containing a set of plausible characters. These characters are considered as the best candidates for the unknown input. Then, a minimal information from the spatial relationship of the unknown input is used to choose the most likely character from the candidates. If the similarity of the candidate characters falls below the similarity threshold, the matching process continues with other characters in neighborhood cells. In addition to improving the recognition of characters, the salient feature space is used to minimize the processing time for sequence matching. With the salient feature vector extracted from the unknown input, the matching process starts at the most likely characters from the feature space where a decision is made in a few steps in the matching process.

4. Experiment

For filtering operations, we use symmetric Gaussian windows of size 5x5 pixels for church documents and 3x3 pixels for ordinary handwritten documents. The larger window size for church documents is due to their noise and thick ink lines. A window size of greater than 3x3 pixel size over-smoothes the thin lines written by ordinary writers.

Because we are extracting the relative size of primitives, the system does not need size normalization of characters. Besides, it does not require training as it relies on a stored knowledge base for recognition. The recognition system is tested on the database, both on the training and test set. There was no difference on the recognition rate of the two sets, as expected. Recognition rate varies depending on the type of document and complexity of characters. Simple shaped characters and those with long primitives such as ሀ, በ, ዘ, ገ, and their derivatives show better recognition accuracy across each group of the database. The average recognition result of each group is summarized in Table 3 below.

Table 3. Recognition result.

Type of document	Recognition rate
Group I	87%
Group IIa	76%
Group IIb	81%

Despite their noise, *Group I* documents show better result because the characters are carefully written and there was a better chance of extracting primitives. The characters in *Group IIa* are more like cursive since they are part of a text, as opposed to *Group IIb* characters which are written isolated. This brought slight difference in the recognition accuracy. For *Group IIa* documents, the reported result is for characters which are not physically connected to others.

5. Conclusion

We presented a writer-independent offline handwriting recognition system and database for Ethiopic script. Since various types of real-world handwritten documents are included in the database, it can be used as a benchmark for testing and comparing character/word segmentation, text line detection, and recognition systems for Ethiopic script. The recognition system does not need training because the knowledge base stores possibly occurring sequences of primitives and connectors for each handwritten character. Since the knowledge base is not built from a specific set of training data and does not depend on writing styles, the system is writer-independent. Size normalization of character images is not required as we encode only the relative length of primitives, which makes the system reasonably size-insensitive. The structural and syntactic analysis efficiently handles neatly and properly written characters. However, such analysis did not yield promising results for characters whose primitives are not connected well, which is commonly appearing in handwritten text. To complement the recognition we use a set of features based on the characteristics of primitives. The recognition system can be further applied to word level recognition by incorporating language models such as lexicon or parts-of-speech tagger.

Acknowledgement: Swedish International Development Agency (SIDA) is gratefully acknowledged for financially supporting this research.

References

[1] Y. Assabie and J. Bigun, “Multifont size-resilient recognition system for Ethiopic script”, *IJDAR*, 10(2): pp. 85-100, 2007.

[2] J. Bigun, *Vision with Direction: A Systematic Introduction to Image Processing and Vision*. Springer, Heidelberg, 2006.

[3] H. Bunke, “Recognition of cursive Roman handwriting: past, present and future”, *Proc. 7th ICDAR*, Edinburgh, Scotland, pp. 448-459, 2003.

[4] C. Huang and S.N. Srihari, “Word segmentation of off-line handwritten documents”, *Proc. Document Recognition & Retrieval XV, IST/SPIE Annual Symp.*, San Jose, 2008.

[5] Y. Li, Y. Zheng, D. Doermann, and S. Jaeger, “A new algorithm for detecting text line in handwritten documents”, *Proc. 10th IWFHR*, La Baule, France, pp. 35-40, 2006.

[6] R. Plamondon, S.N. Srihari, “On-line and off-line hand writing recognition: A comprehensive survey”, *IEEE Trans. PAMI*, 22(1): pp. 63-84, 2000.

[7] S. Selvi and K. Indira, “A novel character segmentation algorithm for offline handwritten character recognition”, *Proc. ICCR*’05, Maysore, India, pp. 462-468, 2005.

[8] S. Uchida and H. Sakoe, “A survey of elastic matching techniques for handwritten character recognition”, *IEICE Trans. Inf. & Syst.*, E88-D(8): pp. 1781-1790, 2005.