Structural and Syntactic Techniques for Recognition of Ethiopic Characters

Yaregal Assabie and Josef Bigun

School of Information Science, Computer and Electrical Engineering
Halmstad University, SE-301 18 Halmstad, Sweden
{Yaregal.Assabie, Josef.Bigun}@ide.hh.se

Abstract. OCR technology of Latin scripts is well advanced in comparison to other scripts. However, the available results from Latin are not always sufficient to directly adopt them for other scripts such as the Ethiopic script. In this paper, we propose a novel approach that uses structural and syntactic techniques for recognition of Ethiopic characters. We reveal that primitive structures and their spatial relationships form a unique set of patterns for each character. The relationships of primitives are represented by a special tree structure, which is also used to generate a pattern. A knowledge base of the alphabet that stores possibly occurring patterns for each character is built. Recognition is then achieved by matching the generated pattern against each pattern in the knowledge base. Structural features are extracted using direction field tensor. Experimental results are reported, and the recognition system is insensitive to variations on font types, sizes and styles.

1 Introduction

Ethiopia is among the few countries in the world which have a unique alphabet of its several languages. The Ethiopic alphabet has been in use since the 5th century B.C.[5] and the present form of the alphabet is obtained after passing through many improvements. At present, the alphabet is widely used by Amharic which is the official language of Ethiopia, and a total of over 80 million people inside as well as outside Ethiopia are using this alphabet for writing.

Research on Ethiopic OCR is a recent phenomenon and there are only few papers presented in conferences [3],[6]. Moreover, it has been difficult to develop a good recognition system for Ethiopic characters due to the complex composition of their basic graphical units. In this paper, we present a novel approach to recognize Ethiopic characters by employing structural and syntactic techniques in which each character is represented by a pattern of less complex structural features called primitives [2] and their spatial relationships. Each character forms a unique set of patterns which are generated from the relationships of primitives. The structural features and their relationships are extracted by using direction field tensor. The characters expressed in terms of primitives and their relationships remain similar under variations on the size, type and style of characters. Accordingly, the present results are novel contributions towards the development of a general Ethiopic OCR system that works independent of the appearance and characteristics of the text.

D.-Y. Yeung et al. (Eds.): SSPR&SPR 2006, LNCS 4109, pp. 118 – 126, 2006.

2 Ethiopic Alphabet

The most common Ethiopic alphabet used by Amharic language has 34 basic characters and other six orders derived from the basic forms making a total of 238 characters. The alphabet is conveniently written in tabular format of seven columns as shown in Table 1, where the first column represents the base character and the other columns represent modifications of the base character.

Table 1. Part of the Ethiopic Alphabet

1st(ä) order	2nd(u) order	3rd(i) order	4th(a) order	5th(e) order	6th(ə) order	7th(o) order
ሀ hä	ሁ hu	ሂ hi	ሃ ha	ሄ he	ህ hə	ሆ ho
ለ lä	ሉ lu	ሊ li	ላ la	ሌ le	ል lə	ሎ lo
ሐ hä	ሑ hu	ሒ hi	ሓ ha	ሔ he	ሕ hə	ሖ ho
⋮	⋮	⋮	⋮	⋮	⋮	⋮

2.1 Structural Analysis

Ethiopic characters are considered to have the most attractive appearance when written with thick appendages, vertical and diagonal strokes, and thin horizontal lines. Most of the horizontal strokes in Ethiopic characters are only a few pixels wide and sometimes they do not exist, especially in degraded documents. Thus, prominent structural features in the alphabet are appendages, vertical and diagonal lines. These prominent features form a set of primitive structures. In this research, we reveal 7 primitive structures which are interconnected in different ways to form a character. Primitives differ from one another in their structure type, relative length, orientation, and spatial position. The classes of primitives are given below.

- **Long Vertical Line (LVL).** A vertical line that runs from the top to bottom level of the character. The primitive is found in characters like ሀ, ሰ, and ዘ.
- **Medium Vertical Line (MVL).** A vertical line that touches either the top or the bottom level (but not both) of a character. ሰ, ከ, and ና are some of the characters that have these primitives.
- **Short Vertical Line (SVL).** A vertical line that touches neither the top nor the bottom level of the character. It exists in characters like ሐ, ቀ, and ሰ.
- **Long Forward Slash (LFS).** A forward slash primitive that runs from the top to the bottom level of a character. It is found in few characters like ሂ, ሥ, and ፖ.
- **Medium Forward Slash (MFS).** A forward slash primitive that touches either the top or the bottom level (but not both) of a character. This primitive is also found in few characters like አ, ፉ, and ጎ.
- **Backslash.** A line that deviates from the vertical line position to the right when followed from top to bottom. The characters ል, ስ, and ሽ have such primitives.
- **Appendages.** Structures which have almost the same width and height. These primitives are found in many characters. Examples are ሂ, ት, and ቹ.

2.2 Spatial Relationships of Primitives

The unique structure of characters is determined by primitives and their interconnection. The interconnection between primitives describes their spatial relationship. A primitive structure can be connected to another at one or more of the following regions of the structure: *top* (t), *middle* (m), and *bottom* (b). The spatial relationship between two primitives α and β connected only once is represented by the pattern αzβ, where z is an ordered pair (x,y) of the connection regions *t*, *m*, or *b*. In this pattern, α is connected to β at region x of α, and β is connected to α at region y of β. Moreover, the primitive α is also said to be spatially located to the left of β. Thus, the spatial relationship is described by *spatial position* (left and right) and *connection region* (t, m, and b).

There may also be two or three connections between two primitives. The first connection detected as one goes from top to bottom is considered as the *principal connection*. Other additional connections, if there exist, are considered as *supplementary connections*. The principal connection between two primitives is an ordered pair formed by the possible combinations of the three connection regions. This will lead to nine principal connection types as represented by the set: {(t,t),(t,m),(t,b),(m,t),(m,m),(m,b),(b,t),(b,m),(b,b)}.

The principal connection (t,t), i.e., two primitives both connected at the top, has five types of supplementary connections: {(m,b),(b,m),(b,b),(m,m)+(b,m),(m,m)+(b,b)}. The principal connection (t,m) has only one supplementary connection: {(b,m)}. The principal connection (m,t) has three types of supplementary connections: {(m,b),(b,m),(b,b)}. The rest principal connections do not have any supplementary connection. This makes up the possibility of two primitives to be connected in 18 different ways: 9 principal connections alone and 9 principal with supplementary connections. This is shown in Table 2 with example characters in brackets.

Table 2. Connection types between two primitive structures

Principal Connection	Supplementary Connections					
	None	(m,b)	(b,m)	(b,b)	(m,m)+(b,m)	(m,m)+(b,b)
(t,t)	(በ)	(ሆ)	(ጭ)	(ዐ)	(ዓ)	(ፀ)
(t,m)	(ሐ)		(ቀ)			
(t,b)	(ስ)					
(m,t)	(ሐ)	(ቀ)	(ፍ)	(ቢ)		
(m,m)	(ዘ)					
(m,b)	(ሡ)					
(b,t)	(ነ)					
(b,m)	(ሣ)					
(b,b)	(ሀ)					

2.3 Representation

Primitives are connected to the left and right of another primitive at one of its three connection regions. To the right of a primitive, two different primitives can also be connected at the middle as in the case of ክ. Therefore, a maximum of three primitives can be connected to the left of another primitive and up to four primitives can be connected to the right. To represent this relationship, a special tree structure having three left nodes and four right nodes is proposed as shown in Fig. 1. Each node in the tree stores data about the type of the primitive itself, the type of connections with its parent primitive, and the spatial positions of primitives connected to it.

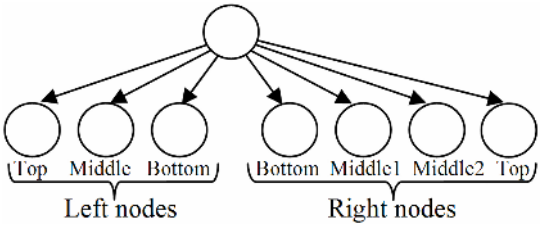

Fig. 1. General tree structure of characters

A primitive is appended to its parent primitive at one of the seven child nodes in the tree based on the principal connection that exists between them. For implementation, primitives are represented by a two digit numerical code as shown in Table 3. The first digit represents their relative length, spatial position and/or structure, and the second digit represents their orientation.

Table 3. Numerical codes assigned to primitive structures

Primitive Types	Numerical Codes
LVL	98
MVL	88
SVL	78
LFS	99
MFS	89
Backslash	87
Appendages	68

Each connection between two primitives is also assigned a two digit number which represents the *left* and *right* spatial positions. The three connection regions, i.e., top, middle, and bottom are represented by the numbers 1, 2, and 3 respectively. For example, using this approach, the connection *(m,b)* is represented by *23*. Connection types with two or more connections between primitives are assigned a numerical code formed by the concatenation of the numerical codes of the respective connections. For example, the numerical code of the connection type *(t,t)+(m,m)+(b,m)* is *112232*. When there is a primitive without being connected to any other primitive in the character, a

connection type of *none* (with code number *44*) is used. In this case, the primitive is appended to one of other primitives based on the closeness in their spatial position. The connection type of the root primitive, which has no any parent, is also *none*.

2.4 Building Primitive Tree and Pattern Generation

The first step in building a primitive tree is identifying the root primitive. Variation in setting root primitive results in a different tree structure which will adversely affect the pattern generated for a character. To build a consistent primitive tree structure for each character, a primitive which is spatially located at the left top position of the character is used as the root primitive. The following recursive algorithm is developed to build primitive tree of characters. The function is initially invoked by passing the root primitive as a parameter.

```
BuildPrimitiveTree (Primitive)
{
    BuildPrimitiveTree(LeftTopPrimitive)
    BuildPrimitiveTree(LeftMidPrimitive)
    BuildPrimitiveTree(LeftBotPrimitive)
    BuildPrimitiveTree(RightBotPrimitive)
    BuildPrimitiveTree(RightMid1Primitive)
    BuildPrimitiveTree(RightMid2Primitive)
    BuildPrimitiveTree(RightTopPrimitive)
}
```

Examples of primitive trees built by the above algorithm are shown in Fig. 2. After building the primitive tree, a string pattern is generated by using in-order traversal of the tree (*left*{top, mid, bottom}, *parent*, *right*{bottom, middle1, middle2, top}). By starting on the root primitive, in-order traversal of the tree generates a unique set of patterns for each character. The algorithm is implemented using a recursive function in a similar way as building the primitive tree.

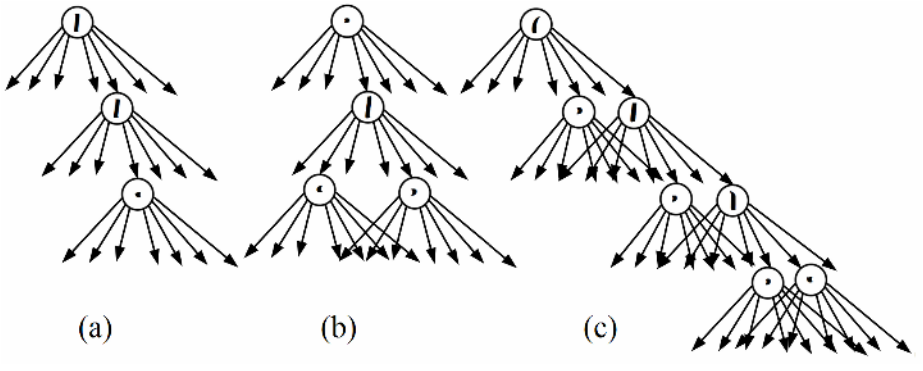

Fig. 2. Examples showing primitive trees for (a) ॥, (b) ঐ, (c) ৬৬

2.5 Alphabet Knowledge Base

The geometric structures of primitives and their spatial relationships remain the same under variations on fonts and their sizes. In Fig. 3*a*, all the different font types and sizes of the character ሀ are described as two Long Vertical Lines both connected at the top. This is represented by the pattern {44,98,11,98}. As there is no structural difference between a Long Vertical Line and its bold version, Fig. 3*b* is also represented by the same pattern as in the case of Fig. 3*a*. In Fig. 3*c*, the character is described as two Long Forward Slashes both connected at the top and it is represented by the pattern {44,99,11,99}. Therefore, any form of the character ሀ is represented as a set of patterns {{44,98,11,98},{44,99,11,99}}. Accordingly, the knowledge base of the alphabet consists of a set of possibly occurring patterns of primitives and their relationships for each character. This makes the proposed recognition technique tolerant of variations in the parameters of fonts.

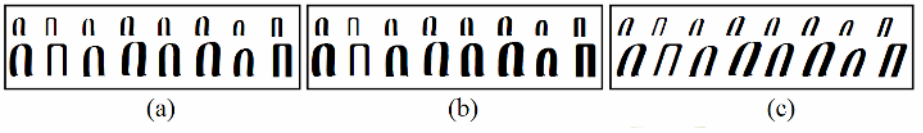

Fig. 3. (a) The Ethiopic character ሀ with different font types and sizes of 12 and 18, (b) bold style of ***ሀ***, (c) italic style of ***ሀ***

3 Extraction of Structural Features Using Direction Field Tensor

A local neighborhood in an image where the gray value changes only in one direction, and remains constant in the orthogonal direction, is said to have Linear Symmetry (LS) property [1]. The LS property of an image can be estimated by analyzing the direction field tensor. The direction tensor, also called the structure tensor [4],[7], is a 3D field tensor representing the local direction of pixels. For a local neighborhood *f(x,y)* of an image *f*, the direction tensor S is computed as a 2x2 symmetric matrix using Gaussian derivative operators D_x and D_y.

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

Linear symmetry exists at edges where there are gray level changes and it can be estimated by eigenvalue analysis of the direction tensor using complex moments of order two which are defined as follows.

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

$$I_{11} = \iint \left| (D_x + iD_y) f \right|^2 dxdy \tag{3}$$

The complex partial derivative operator $D_x + iD_y$ is defined as:

$$D_x + iD_y = \frac{\partial}{\partial x} + i\frac{\partial}{\partial y} \tag{4}$$

The value of I_{20} is a complex number where the argument is the local direction of pixels in double angle representation and the magnitude is a measure of the local LS strength. The scalar I_{11} measures the amount of gray value changes in a local neighborhood of pixels. Direction field tensor, which is a 3D tensor field, can also be conveniently represented by the 2D complex I_{20} and 1D scalar I_{11}. The complex image I_{20} can be displayed in color as shown in Fig. 4 where the hue represents direction of pixels in double angle representation.

Due to the Gaussian filtering used in the computation of direction tensor, the LS strength (magnitude) at the orthogonal cross-section of edges in the image forms a Gaussian of the same window size. Therefore, the cross-section of lines in the I_{20} image can be reduced to a skeletal form (one pixel size) by taking the point closest to the mean of the Gaussian formed by the LS strength in the orthogonal direction. The skeletal form of I_{20} image is then used for extraction of structural features.

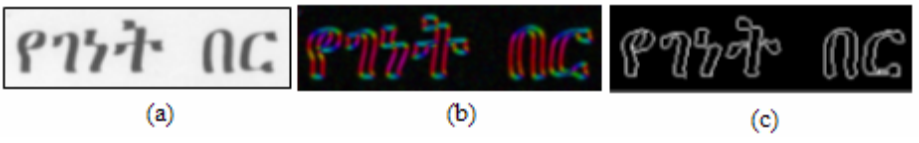

Fig. 4. (a) Scanned document, (b) I_{20} of $\boldsymbol{a}$ where hue represents direction, (c) skeletal form of $\boldsymbol{b}$ without direction information

Before extracting structural features, characters are segmented into individual components. In the skeletal form of the I_{20} image, horizontal spaces that lack LS (LS strength <0.05 after normalization) are used to segment text lines, and vertical spaces that lack LS are used to segment characters in the text lines. Rectangular boxes in Fig. 5 show segmented characters of Ethiopic text. Since the direction of pixels is represented by double angle, the angle θ obtained from the argument of I_{20} is in the range of 0 to 180 degree. The direction of pixels at the edges of primitives is close to 0 and 180 degrees and can be converted to the range of 0 to 90 degrees by ε=abs(90-θ) so that ε for primitives is consistently close to 90 degree. In this study, pixels with $\varepsilon >= 30^{o}$ and having strong LS property (LS strength >=0.05) are considered as parts of primitives, and those with $\varepsilon < 30^{o}$ and having strong LS property are considered as parts of connectors. A primitive in the grayscale image will have

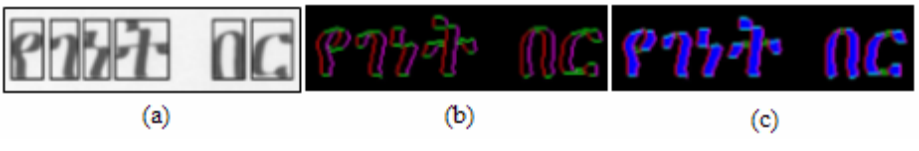

Fig. 5. (a) Character segmentation mapped to the original image, (b) skeletal form of the I_{20} image where the red and purple colors show the left and right edges of primitives respectively, and the green color shows connectors, (c) extracted primitives shaded with blue color

two lines (left and right edges) in the skeletal form of the I_{20} image. Primitive structures are then constructed by the two matching lines. The group information about direction and spatial position of pixels in a primitive are finally used to classify the primitives.

4 The Recognition Process

A general recognition system of Ethiopic characters is proposed as shown in Fig. 6. Characters are segmented by making use of direction field tensor. Structural features are then extracted and a pattern of their spatial relationships is generated for each segmented character. A character is said to be recognized if the string pattern generated from primitive tree has a matching pattern in the knowledge base. Pattern matching is done by comparing the string pattern generated from the image against with each string pattern stored in the knowledge base. The similarity of each comparison is computed and the most similar pattern is considered to decide whether the string pattern is recognized or not. This is done by setting a threshold of similarity.

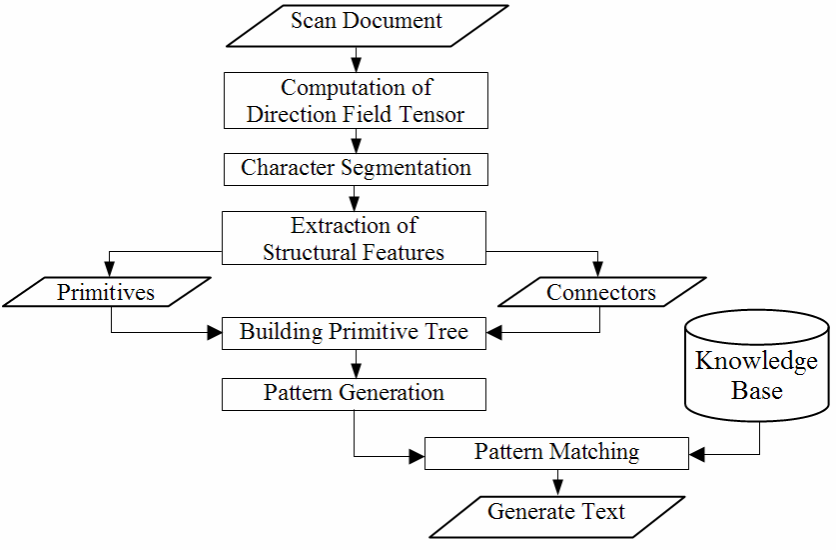

Fig. 6. Flowchart of the recognition process

5 Experiment

The size of the Gaussian window used for filtering operations is determined by the size of fonts in the document. For example, a Gaussian window of pixel-size 3x3 was efficient for texts with font size of 12, and a window size of 5x5 was found to be better for font sizes of 18.

There is no standard image database of Ethiopic text developed for testing character recognition systems. Thus, the experiment was done on images of about

thirty pages scanned from newspapers, books and clean printouts that contain characters of different fonts and sizes varying from 12 to 18. Images taken from clean printouts show better recognition due to their relatively better quality. A recognition rate of 92% was achieved for clean printouts and the recognition accuracy for newspaper and books was 86%. However, there was no difference in recognition accuracy due to variation in fonts, and larger font sizes tend to be recognized slightly better than their smaller versions.

The structural and syntactic method used for recognition of Ethiopic characters is efficient to uniquely identify the characters. Recognition errors mainly come from poor quality of documents. Character segmentation errors also affect the overall character recognition accuracy. The other process that hampers the recognition process is extraction of connectors. This is due to the fact that horizontal lines in Ethiopic characters are very thin and sometimes absent especially in degraded and low quality documents. The algorithm used to extract primitives works well even in noisy documents.

6 Conclusion and Future Work

In this paper, a novel approach is proposed for Ethiopic character recognition. Structural and syntactic techniques are effectively used to uniquely represent the complex structure of characters by the relationships of less complex primitive structures. To this end, direction field tensor is used as a tool for extraction of structural features. The use of Gaussian separable filters to compute direction field tensor made the computation time minimal. The recognition accuracy can still be improved to a higher level by working more on character segmentation, extraction of structural features and pattern matching algorithms. Extraction of structural features can be further improved by applying statistical techniques. The process of pattern matching and classification is expected to perform better by using neural networks. In general, the recognition system is insensitive to variations on the size, type and other parameters of characters and therefore, the overall research activity will lead to the development of efficient OCR software for Ethiopic script.

References

1. J. Bigun, *Vision with Direction: A Systematic Introduction to Image Processing and Vision*, Springer, Heidelberg, 2006.
2. H. Bunke and A. Sanfeliu, *Syntactic and Structural Pattern Recognition: Theory and Applications,* World Scientific, Singapore, 1990.
3. J. Cowell and F. Hussain, "Amharic character recognition using a fast signature based algorithm," *Proc. Fourth Int'l Conf. Information Visualization*, 2003, pp. 384-389.
4. W. Forstner, "A framework for low level feature extraction", in J. Eklundh (ed.), *Lecture Notes in Computer Science*, vol. 801, Springer-Verlag, Berlin, 1994, pp. 383-394.
5. A. S. Gerard, *African Language Literatures: An introduction to the literary history of sub-Saharan Africa*, Three Continents Press, Washington, 1981.
6. L. Premaratne, Y. Assabie and J. Bigun, "Recognition of modification-based scripts using direction tensors," *ICVGIP'04*, Kolkata, 2004, pp. 587-592.
7. J. Weickert, "Coherence-enhancing shock filters," in B. Michaelis and G. Krell (eds.), *Lecture Notes in Computer Science*, vol. 2781, Springer-Verlag , Berlin, 2003, pp 1-8.