OFFLINE HANDWRITING RECOGNITION USING PARAMETER VECTOR ANALYSIS

Sujal Raul¹, Nikita Shah², Akshay Kulkarni³ and G. Anuradha⁴

¹,²,³,⁴St. Francis Institute of Technology, Mumbai, India, ¹E-mail: sujal.raul@yahoo.co.in ²niki.shah1711@gmail.com ³kulkarni_aks007@yahoo.co.in, ⁴ganusrinu4@yahoo.co.in

Abstract: Handwriting recognition is a process which translates handwritten documents into digital data that can be recognized by a computer. There are two general types of handwriting recognition: Offline recognition and online recognition. This paper proposes a new approach for offline handwriting recognition. In this approach we recognize handwritten characters using a technique “Parameter Vector Analysis”. Each character can be viewed as a geometrical structure made up of lines and curves. By analyzing the pictorial view of each character, a parameter vector can be generated which will be unique for each character. This recognition process is built as supervised learning and recognition of character is based on predefined parameter vectors for all the characters in a language.

Keywords: Handwriting Recognition, Parameter Vector Analysis, Parameter Vector, Co-ordinate Geometry, Vector Space, Image Component Vector, Angle Component Vector.

1. RELATED WORKS

Handwriting recognition has been one of the most fascinating and challenging research areas in the field of image processing and pattern recognition in the recent years. It contributes immensely to the automation of various tasks between man and machine in numerous applications. Handwriting recognition is the ability of a computer to receive and interpret handwritten input from sources such as paper documents, photographs, touch-screens and numerous other devices. In general, handwriting recognition is classified into two types as off-line and on-line handwriting recognition methods. In the off-line recognition, the writing is usually captured optically by a scanner and the completed writing is available as an image. But, in the on-line system the two dimensional coordinates of successive points are represented as a function of time and the order of strokes made by the writer are also available. The on-line methods have been shown to be superior to their off-line counterparts in recognizing handwritten characters due to the temporal information available with the former. Several applications including mail sorting, bank processing, document reading and postal address recognition, scanning different types of forms and automatically updating corresponding databases require off-line handwriting recognition systems. As a result, the off-line handwriting recognition continues to be an active area for research towards exploring the newer techniques that would improve recognition accuracy. The central tasks in offline handwriting recognition are character recognition and word recognition. A necessary preliminary step to recognition written language is spatial issue of locating and registering the appropriate text when complex, two-dimensional spatial layouts are employed. Handwriting is a skill that is personal to individual. There exist enumerable numbers of styles in which a character may be written. Due to this fact handwriting recognition is a tough problem and desirable accuracy is very difficult to achieve. The solutions being proposed mainly use Artificial Neural Networks (ANN) [1], Hidden Markov Models (HMM) [2] and Genetic algorithms for character recognition [3]. Then when an unknown input is given to the system, the Artificial Neural Network is able to find out the most probable character by generalization. Hence once trained, the system would be ready to recognize the given unknown input. Hidden Markov Model is a complete
statistical model that tries to predict the unknown sequence. Hence it also tries to recognize the unknown character which is given as input. Finally the genetic algorithm based approach tries to match the input to the training data and the data generated from intermixing of training data, to find the best match for the given input data. In all the above methods some artificial agent (i.e. ANN, Character Graph Generator) exists. In this method of character recognition no external artificial agent is required. Based on the shape and view of each character parameter vector is constructed and recognition is performed using parameter vector.

2. STEPS FOR OFFLINE HANDWRITING RECOGNITION

Handwriting recognition is a famous problem which involves the recognition of whatever input is given in form of image, scanned paper, etc.

A. Preprocessing: This step involves the initial processing of the image like thresholding, converting gray scale image to binary black and white image, noise removal, extraction of the textual matter; so that it can be used as an input for the recognition system. The individual characters are written discretely instead of cursively.

The task of thresholding foreground ink from the background paper. The basic idea of thresholding is by using the histogram of gray scale values of the character image. [7]

For handwritten documents the connectivity of the strokes has to be preserved. Digital capture of images can introduce noise from scanning devices and transmission media. Smoothing operations are used to eliminate the artifacts introduced during image capture. One study [8] describes a method that performs selective and adaptive stroke filling with neighborhood operator which emphasizes stroke connectivity. But in this method, a common problem which is encountered is the interference of strokes from neighboring text lines. So one approach discussed in [9] is to follow strokes in thinned images to segment the interfering strokes from the signal.

B. Segmentation: This step deals with the breaking of the lines, words and finally getting all the characters separated. This step involves the identification of the boundaries of the character and separating them for further processing. In this algorithm we assume that this step is already done. Hence the input to our system is a single character.

C. Recognition: Once the input image is available in good condition, it may be processed for recognition.

The role of the recognition system is to identify the character. The proposed algorithm uses an image as an input for the same.

In the case of print image the problem of assigning a digitized character to its symbolic class is called Optical Character Recognition (OCR) [10]. In the case of handprint this process is referred to as Intelligent Character Recognition (ICR). The typical classes used in our proposed algorithm are upper case characters of English language and ten digits in its handwritten form.

3. HANDWRITTEN CHARACTER RECOGNITION BASED ON “PARAMETER VECTOR ANALYSIS (PVA)”

Each character is made up of a number of strokes and curves. Hence strokes (or segments) and curves are constructing elements of a character. We try to analyze this pictorial view of a character and “A Parameter Vector” for each character is build up.

3.1 Components of Parameter Vector

For character ‘c’ in a language parameter vector is defined as:

\[ \text{PV}_c = [\text{V}_1^c, \text{V}_2^c] \]

Where

\[ \text{V}_1^c = [\text{BPC}, \text{IPC}, \text{SGC}, \text{CC}] \]
\[ \text{V}_2^c = [\text{Virtual Boundary Inclined Angles}] \]

BPC - Boundary Point Count
IPC - Intersection Point Count
SGC - Segment Count
CC - Curve Count
3.2 Components of Image Component Vector \([V_{1C}]\)

Boundary Point Count (BPC): It indicates total number of boundary points present in a character. Boundary Point is an extreme point in the character curve which has a “SINGLE” neighboring pixel.

Intersection Point Count (IPC): It indicates total number of intersection points present in a character. Intersection Point is an intersection point of any two or more constructing elements.

Segment Count (SGC): It indicates a count of all segments or strokes that appear in a character.

Figure 1: Image Component Vector for “A”

Curve Count (CC): It indicates a count of all curves that appear in a character.

Figure 2: Image Component Vector for “P”

3.3 Components of Angle Component Vector \([V_{2C}]\)

From given input image each character is segmented, preprocessed and is made available to recognition process. The shape of each character is unique; the angle subtended by boundary segments (segments that pass through extreme points of a character). In order to calculate this angle, virtual boundary is created that passes through all the extreme points on a character in all four directions (bottom, right, top, left). Virtual Boundary for character ‘Q’ is constructed as shown in figure 3.

Assuming bottom left point to be the initial point, construct a virtual boundary in an anticlockwise direction. This virtual boundary will pass through all the extreme points on a character. Identify all the segments that pass through extreme point and for each segment calculate “Approximate Angle of Inclination” with respect to virtual boundary. If there exist a segment parallel to virtual boundary surface, Angle of Inclination is assumed to be zero.

\[ V_{2c} = [(I_b) (I_r) (I_t) (I_l)] \]

Where

- \( I_b \) = angle of inclination with virtual boundary bottom surface
- \( I_r \) = angle of inclination with virtual boundary right surface
- \( I_t \) = angle of inclination with virtual boundary top surface
- \( I_l \) = angle of inclination with virtual boundary left surface

If more than one angle exists, then all angle must be included (Separated by comma)

Length of Virtual Boundary Inclination Vector \((V_{2c})\) is,

\[ \text{Len} (V_{2c}) = \text{Total number of boundary surfaces for which angle of inclination exists.} \]
Length of $V_2$ will vary depending on a character and number of boundary segments for that character.

$$V_2[A] = [(I_b) (I_r) (I_t) (I_l)] = [(50) (40) (60) (-)]$$

$$V_2[P] = [(I_b) (I_r) (I_t) (I_l)] = [(90) (10) (20) (0)]$$

3.4 Flow Chart of Parameter Vector Analysis (PVA)

The flowchart given below illustrates the prior steps that need to be taken before the parameter vector is being generated. Initially the input image needs to be expand to the given pixel size of application so that thinning algorithm can be applied to it. After the unit pixel image is obtained we need to walkthrough the character to identify parameters like intersection points, lines/curves, boundary points, segments and curves. Once this is done the respective parameter vector can be generated.

Table 1

<table>
<thead>
<tr>
<th>CHARACTER</th>
<th>PARAMETER VECTOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>[2 3 5 0] AND [(50) (40) (60) (-)]</td>
</tr>
<tr>
<td>B</td>
<td>[0 3 2 2] AND [(0) (10) (0) (0)]</td>
</tr>
<tr>
<td>C</td>
<td>[2 0 0 1] AND [(10) (10) (30) (30)]</td>
</tr>
<tr>
<td>D</td>
<td>[0 2 0 1] AND [(5) (10) (0) (0)]</td>
</tr>
<tr>
<td>E</td>
<td>[3 3 6 0] AND [(0) (90,90) (0) (0)]</td>
</tr>
<tr>
<td>F</td>
<td>[3 2 4 0] AND [(90) (90) (0) (0)]</td>
</tr>
<tr>
<td>G</td>
<td>[2 2 2 1] AND [(0) (0,5) (10) (15)]</td>
</tr>
<tr>
<td>H</td>
<td>[4 2 5 0] AND [(90) (90) (0) (0)]</td>
</tr>
<tr>
<td>I</td>
<td>[2 0 1 0] AND [(90) (90) (0) (0)]</td>
</tr>
<tr>
<td>J</td>
<td>[2 1 1 1] AND [(5) (0) (90) (-)]</td>
</tr>
<tr>
<td>K</td>
<td>[4 1 4 0] AND [(90) (35) (40) (0)]</td>
</tr>
<tr>
<td>L</td>
<td>[2 1 2 0] AND [(0) (90) (-) (0)]</td>
</tr>
<tr>
<td>M</td>
<td>[2 2 3 0] AND [(90,45) (0) (45) (0)]</td>
</tr>
<tr>
<td>N</td>
<td>[2 2 3 0] AND [(90) (0) (90) (0)]</td>
</tr>
</tbody>
</table>

Table Cont’d
The above table calculates Parameter Vector (PV) value for standard letters A-Z. These values are calculated using the procedure explained PVA flowchart.

Step 1: Initial database population
Assign “VS” for each character ‘c’ in a language. (Size of “VS” will increase dynamically. New PV’s will be added to VS of a character ‘c’ whenever an unknown input is recognized as ‘c’.)

For each character c
For each character style ‘cs’
Population database “Vector Space” with parameter vector
End
End

Step2: From unknown input image segmented and preprocessed characters are received. Generate a parameter vector for each unknown input as explained in previous section of this paper.

Step3: Matching criteria Validation:
Let PVc be the unknown character parameter vector. Let PVk be the known character parameter vector. Value of PVk is taken from table 1 for characters and from table 2 digits.

Condition 1:
Observed calculations show that length of Virtual Boundary Inclination Vector for unknown input (V2c) should be nearly equal to the Virtual Boundary Inclination Vector for known character (V2k). Hence we assign a threshold of ‘1’ to this criterion.

| Len(PVc) – Len(PVk) | < = 1

Condition 2:
If condition1 is satisfied then Evaluation Vector is defined as

EVck = V1ck AND V2ck

Where,

V1ck = [Mod (BPCc - BPCk) Mod (IPCc - IPCk) Mod (SGCc - SGCK) Mod (CCc - CCk)]

V2ck = [Mod (Ibc - Ibk)]
\[ \text{Mod} (I_{c - I_{rk}}) \]
\[ \text{Mod} (I_{c - I_{rk}}) \]
\[ \text{Mod} (I_{c - I_{rk}}) \]

If more than one angle exists for a single surface, mod of each angle of \( V_{2c} \) with each angle of \( V_{2k} \) must be calculated.

\( EV_{ck} \) matches recognition criteria if,

1: Not more than two parameters of \( V_{1ck} \) exceed the difference “2”

AND

2: \( V2ck \) for each boundary surface must satisfy a threshold of “20 degrees”

If more than one angle exists for a single surface, then at least one Mod () operation should result in a specified threshold.

3: Above values have been calculated from heuristic results for sample data set.

If evaluation vector satisfies above two conditions, unknown input character ‘C’ is recognized as Character ‘K’ in a language.

Step 4: Vector Space Modification
Parameter Vector of ‘C’ is added to the vector space of character ‘K’

(This step is repeated until desired accuracy is obtained. Once the database is populated with sufficient number of parameter vectors, this step can be skipped.)

Step 5: Final Conversion Recognized character is converted into machine readable ASCII format.

5. WORKING OF ALGORITHM

\[ PV_c = [3 1 1 2] \text{ AND } [(25) (15) (25) (25)] \]
\[ \text{Len} (PV_c) = 4. \]

Now we refer table 2 for comparison. Hence matching criteria validation eliminates comparison with \( PV_1 \) and \( PV_7 \).

While checking for condition 2 in validation criteria we find that, \( PV_3 = [3 1 0 2] \text{ AND } [(10) (30, 30) (40) (140, 90, 10)] \)

\[ EV_{c3} = \text{Mod}(3-3) \text{ mod}(1-1) \text{ mod}(1-0) \text{ mod}(2-2) \]
\[ \text{AND} \]
\[ \text{Mod}(25-10) \text{ Mod}(15-30) \text{ Mod}(15-30) \text{ Mod}(25-40) \text{ Mod}(30-140) \text{ Mod}(30-90) \text{ Mod}(25-10) \]

\[ EV_{c3} = [0 0 0 0] \text{ AND } [(15) (15) (15) (15)] \]

Hence unknown input is recognized as ‘3’ and its parameter vector is added to Vector space of ‘3’.

Similarly, for alphabets identification table 1 can be used.

6. CONCLUSION

In this paper we address the problem of Offline Character Recognition by proposing an algorithm PVA based on geometrical shapes of characters. The database is manually created for characters (A–Z) and digits (0–9). When the algorithm is implemented it will yield the same database. When an unknown input is given to the Algorithm, the Image Component Vector and Angle component Vector for the same are generated which are then compared with the existing database. Initially, the length of vectors is compared with the Principle Vectors generated for character set and digits. The values in Principle Vectors are associated with Threshold values which are also considered during comparison. If the values are within the Threshold range the characters will be identified else rejected. This proposed algorithm helps in generating a gene pool which can be further used for optimization purposes in handwritten character recognition applications.

REFERENCES


