2012 3rd International Conference on Information Security and Artificial Intelligence (ISAI 2012)
IPCSIT vol. 56 (2012) © (2012) IACSIT Press, Singapore
DOI: 10.7763/IPCSIT.2012.V56.30

Preprocessing of Gurmukhi Strokes in Online Handwriting Recognition

Nainsi Gupta [1+], Mayank Gupta [2] and Rahul Agrawal [2]

[1] M.Tech Student, Dept. of Computer Science and Engineering, SRCEM,Gwalior

[2] M.Tech Student, School of Mathematics and Computer and Applications, Thapar University, Patiala

Abstract. The pen-based devices are not so common in India and one reason for that is the absence of localized applications. Developing an online handwriting recognition system for Gurmukhi scripts for such devices would play an important role in making these devices available and usable for the Indian society. Preprocessing will play an important role in increasing the accuracy of such online handwriting recognition system. A model for preprocessing a Gurmukhi stroke is proposed. This model consists of 5 basic algorithms for preprocessing. Prior to these algorithms, a basic step called Stroke Capturing is done, which samples data points along the trajectory of an input device (electronic pen or mouse) while the character is drawn.

Keywords: Online Handwriting Recognition, Preprocessing, Gurmukhi Strokes.

1. Introduction

In current era, a new technology called pen computing is emerging. It includes mobile devices and applications in which electronic pen with a writing pad is used as the main input tool. For implementing pen-computing applications, online handwriting recognition system should be used. Online handwriting recognition systems have been developed for various scripts around the world. Very few attempts have been made to build an online handwriting recognition system for Gurmukhi script. Handwriting recognition is the task of transforming a language represented in its graphical form into symbolic representation. Handwriting recognition systems are broadly divided into two: namely **offline and online handwriting recognition systems**. A typical online handwriting system shows in Figure 1, which shows four main parts of online handwriting recognition:

Data acquisition: It is the first phase in online handwriting recognition that collects the sequence of coordinate points of the moving pen.

Preprocessing: Preprocessing phase in handwriting recognition is applied to remove noise or distortions present in input text due to hardware and software limitations. In online handwriting recognition, preprocessing includes five common steps, namely, size normalization and centering, interpolating missing points, smoothing, slant correction and re sampling of points.

Feature extraction: Feature extraction is essential for efficient data representation and for further processing. Also, high recognition performance could be achieved by selecting suitable feature extraction method.

Recognition: According to extracted features, compared with reference stroke, decision can be made through some kind of decision rules.

+ Corresponding author. Tel.: +91-9575764432
E-mail address: naincy_negupta@yahoo.com, mayankgupta9854@gmail.com, rahul.thapar021@gmail.com..

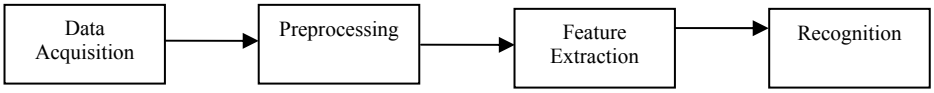

Fig. 1: Phases of online handwriting recognition.

The presence of online handwriting recognizer for Devanagari, Gurmukhi, Bangla and other Asian scripts shall provide a natural way of communication between users and computers and it will increase the usage of personal digital assistant or tablet PCs in Indian languages. Also, some of the Asian scripts such as Devanagari, Gurmukhi, Bangla and Tamil share many similarities and therefore advances made for one script with respect to online handwriting recognition could be useful for other such similar scripts. This paper has been done using strokes written in Gurmukhi script. Detailed description of Gurmukhi script is given in the next section.

2. Gurmukhi Script

The word "Gurmukhi" literally means "from the mouth of guru". Gurmukhi script is used primarily for the Punjabi language, which is the world's 14th most widely spoken language [1]. Gurmukhi characters are shown in Fig. 2. Gurmukhi script is written in left to right direction and in top down approach [2]. Most of the Characters have a horizontal line at upper part. The characters of words are connected mostly by this line called head line.

ੳ, ਅ, ੲ, ਸ, ਹ, ਕ, ਖ, ਗ, ਘ, ਙ, ਚ, ਛ, ਜ, ਝ, ਞ, ਟ, ਠ, ਡ,
ਢ, ਣ, ਤ, ਥ, ਦ, ਧ, ਨ, ਪ, ਫ, ਬ, ਭ, ਮ, ਯ, ਰ, ਲ, ਵ, ੜ, ਸ਼,
ਖ਼, ਗ਼, ਜ਼, ਫ਼, ਲ਼

Fig. 2: Gurmukhi Characters.

3. Stroke Capturing, Preprocessing algorithms for Gurmukhi Strokes

The important phases in an online handwriting recognition system before recognition process are stroke capturing, preprocessing.

3.1. Stroke Capturing, Preprocessing algorithms for Gurmukhi Strokes

In this phase, the data has been collected in the form of co-ordinates. These co-ordinates are input by mouse or writing pads by using stylus/pen. The sequential positions of the stroke are stored dynamically.

The Samples of i^{th}stroke$S_i = \{(X_{io}, Y_{i0}), (X_{i1}, Y_{i1}), \dots, (X_{io}, Y_{i0}), \dots, (X_{in}, Y_{in})\}$ expressed by the time sequence of co-ordinates as Figure 3 represents the generation of the data for a sample of a Gurmukhi characters in which (X_0, Y_0) and (X_n, Y_n) denote the initial and the final point of the generated time sequence of co-ordinates for the particular sample.

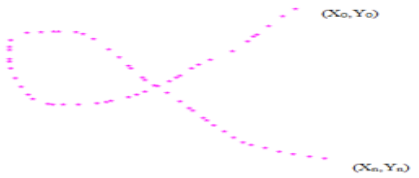

Fig. 3: Stroke of a Gurmukhi character.

3.2. Preprocessing of Strokes

Preprocessing is the most important phase in an online handwriting recognition system. The main purpose of preprocessing phase in handwriting recognition is to remove noise or distortions present in input text due to hardware and software limitations and convert it into a smooth handwriting. These noise or distortions include different size of text, missing points in the stroke during pen movement, jitter present in stroke, left or right slant in handwriting and uneven distances of points from adjacent positions.

In our system the writing area contains a window of size 300×300 pixels. Writer will be able to write inside the window. In the preprocessing phase of online handwriting recognition system under consideration,

we have given mainly five steps. These are size normalization and centering of stroke, interpolating missing points in stroke, smoothing of stroke, uniformity of points in a stroke and resampling of points in stroke [3].

3.3. Size Normalization and Centering of Stroke

Size normalization depends on how writer moves the pen on writing pad. Centering is required when the writer moves pen along the boundary of writing pad [4]. Stroke is not generally centered when the pen is moved along the boundary of writing pad. Size normalization and centering of stroke is a necessary process that should be performed in order to recognize a stroke. The algorithm for size normalization and centering of stroke is given on next page.

Algorithm 3.1: Normalization of stroke.
Step 1: Calculate the co-ordinates of window enclosing the stroke.
Step 2: Find the height (Wyd) and width (Wyd) of window that was calculated in step 1.
Step 3: If $Wxd = 0$ or $Wyd = 0$ then make both Wxd and Wyd equal to1.
Step 4: Compute midpoint ($midx, midy$) of the window enclosing the complete Gurmukhi stroke calculated in step1.
Step 5: Find aspect ratio of the Gurmukhi stroke's window as the ratio of Wxd and Wyd.
Step 6: Now apply transformation matrix to the points in the stroke to perform size normalization and centering of stroke. This transformation matrix is a composite matrix of window to viewport transformation; it consists of translation of window enclosing the stroke, to origin, scaling it to the size of viewing window i.e., 300×300 and finally translating it back to the centre of the viewing window.
Step 7: Exit.

3.4. Interpolation

High speeds of handwriting or hardware limitations can results into missing points. These missing points can be interpolated using various techniques such as Bezier interpolation [5]. In present study, piecewise Bezier interpolation has been used, because it helps to interpolate points among fixed number of points. In piecewise interpolation technique, a set of consecutive four points is considered for obtaining the Bezier curve. The next set of four points gives the next Bezier curve [6].

3.5. Uniformizing of points

Uniformizing of points in a stroke is required to remove excessive points that are aggregated at a particular point in a stroke while writing. In this algorithm only those points are taken into consideration that has distance between them greater than a particular threshold value. Algorithm used for making points uniform in a stroke is given below. At the end of this algorithm list U will contain all uniform points and $count$ will contain number of points.

Algorithm 3.2: Uniformizing of Points
Step 1: Create an empty list U for storing the points generated after by this algorithm.
Step 2: Let n be the number of points currently in the stroke.
Step 3: Let D be the threshold value (distance between adjacent points) on the basis which uniform algorithm is performed.
Step 4: Set $j = 2$ and $count = 0$.
Step 5: Put the points P_1 in the list U and increment count by 1.
Step 6: Repeat steps (i) to (iv) for $i = 1, 2, 3, \ldots, n$
- (i) Calculate distance ($dist$) between P_i andP_j.
- (ii) If $dist \geq D$ then
  - (a) Put P_j in the list U and increment count by 1.
  - (b) Assign j to i.
- (iii) End if.
- (iv) Increment$j$ by 1.

Step 7: Exit.

3.6. Smoothing

Flickers exist in handwriting because of individual handwriting style and the hardware used [7]. Figure 4 shows how 2-neighbors from each side can be considered for this purpose. In this figure five points of the list, generated in the previous step, have been used for smoothing of the stroke. The point P_i has been modified

or smoothes with the help of points $P_{(i-2)}, P_{(i-1)}, P_{(i+1)}, P_{(i+2)}$.If we consider three points then it will not affect the nature of stroke and if we consider more than five points then we tend to lose the nature of stroke in terms of sharp edges.

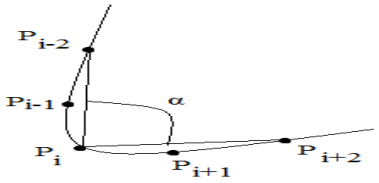

Fig. 4: Formation of angle α at point P_i

3.7. Resampling of Stroke

When the writer is writing slowly, more number of points will be present in the stroke than when the writer is writing quickly. To get rid of such variations resampling is applied. Resampling is the process of resampling the data such that the distance between adjacent points is approximately equal. Besides the removal of the variation, this step essentially reduces the number of points in a stroke, to a desired value. After resampling, the data is significantly reduced and the irregularly placed data points that create jitter on the trajectory of the stroke are removed. This makes the resampling step very useful in noise elimination as well as data reduction. In this phase new data points are calculated on the basis of the original points of list. After this phase, only 64 equidistant points will be present in the stroke.

Algorithm 3.3: Resampling a Stroke

Step 1: Create an empty list R which will contain for storing the points generates after this algorithm.
Step 2: Let S be the list containing all the points processed before resampling algorithm.
Step 3: The number of points currently present in the stroke is n.
Step 4: If $n \leq 64$, then Exit.
Step 5: Else
(i) Find a factor (f) as a ratio of n and 64, only 64 points are desired to be present in the stroke.
(ii) Calculate the resampled points from the list S as $S_1, S_{(1+f)}, S_{(1+2f)}, \ldots, and\ so\ on$ until whole list S is traversed.
(iii) Insert these points in the new list R and mark them as processed points in list S.
(iv) If list R contains 64 points then Exit.
(v) Else
(a) Calculate a new factor as the ratio of unprocessed points in list S and number points less than 64 in list R.
(b) Calculates remaining points using new factor starting from first unprocessed point in the list S.
(vi) End if.
Step 6: End if.
Step 7: Exit.

In the present study, these 5 algorithms have been used. In this work uniform algorithm has been used twice *i.e.*, in step 3 and step 6. In these steps different values of the threshold D (described in Algorithm 3.2) has been used. In step 3, parameter D has value 10 and in step 6 it has value 5. Similarly the interpolation phase is also used two times while performing preprocessing i.e. in step 2 and step 4. In step 1and step 5 normalization and smoothing has been used, and resampling is done at the end of the preprocessing phase. In this system, points generated after preprocessing are used as a feature for recognition. These points are always fixed and are equidistant as far as possible. Number of points in a stroke is fixed to 64.The next section covers discussions and results.

4. Discussions and Results

The algorithms discussed above have been implemented for preprocessing of strokes written in Gurmukhi.100 different words were written by 3 writers. The strokes of these words are then preprocessed using all the steps of preprocessing. Each word written by three different writers, each word consists of a number of strokes and each stroke contains a number of points. The main purpose of the preprocessing in the

current system is to generate a stroke that contains 64 equidistant points. Table 1 gives success rate of the preprocessing phase proposed in this work, for three different writers.

Table 1: Success rate for each writer

Writers / Strokes	Writer 1	Writer 2	Writer3	Total
Non-preprocessed strokes (100 words)	810	747	876	2433
Preprocessed strokes with 64 points	726	702	769	2197
Success Rate (in %)	89.50	93.97	87.78	90.30

Table 1 gives writer–wise success rate for preprocessing phase. It has been observed that more strokes of writer 2 has been successfully preprocessed with 64 points present in them as compared to writer1 and writer 3. Preprocessing algorithms have a least success rate for writer 3. Preprocessing algorithms have a least success rate for writer 3. Overall success rate of the system for three writers is 90.30%.

Table 2 gives number of strokes that our system is unable to preprocess successfully. Three ranges hase been decided according to number of points present in the preprocessed stroke per writer.

Table 2: Number of strokes without 64 points after preprocessing

Preprocessed strokes	Writer1	Writer2	Writer3
with 60-63 points	17	5	20
with 50-59 points	26	2	34
with < 50 points	41	38	53
Total (without 64 points)	84	45	107

Table 2 gives writer-wise number of strokes that are not preprocessed successfully. Three ranges have been defined for unsuccessfully preprocessed strokes, *i.e.*, preprocessed strokes with points in the range 60 to 63, strokes with points in the range 50-59 points and stroke with less that 50 points. This shows that **51.2%, 84.4% and 49.5%** of strokes contain less 50 points out of total unsuccessfully preprocessed stroke for writer 1, writer 2 and writer 3 respectively.

It is worth mentioning that 52 unique strokes are present in 100 words considered in this work. These words were written by three different writers. The preprocessing steps proposed in this paper are not able to correctly preprocess only 3 strokes out of these 52 strokes, thus achieving a success rate of 94.20%.

5. Conclusions

We have introduced a new preprocessing scheme for online handwritten Gurmukhi stroke, which generates the pattern in a manner that creates similarity between different occurrences of the same character. Steps that are included in this preprocessing scheme are: Size Normalization and centering, Interpolation, Uniformizing, Interpolation, Smoothing, Uniformizing, Resampling.After preprocessing of strokes that are present in 100 words written by three different writers, it was found that out of 52 uniques stroke only 5 strokes were incorrectly preprocessed. Only 3 strokes out of these 5 were critical (error rate more that 50%), thus giving a success rate of 94.20%. If we talk about overall accuracy of the preprocessing, out of 2433 strokes, 2197 strokes were correctly preprocessed attaining a success rate of 90.30%.

6. Acknowledgements

The authors would like to gratefully acknowledge **Prof. Manoj Ramaiya** and **Dr. R.K. Sharma** for providing data, feedback for this research.

7. References

[1] A. Sharma. Online handwritten Gurmukhi character recognition, Ph.D. thesis, School of Mathematics and Computer Applications, Thapar University, Patiala, 2009.

[2] P. Singh and N. Tyagi.Radial Basis Function For Handwritten Devanagari Numeral Recognition.*International Journal of Advanced Computer Science and Applications*.2011, vol. 2, no. 5

[3] S. Jaeger, S. Manke, J. Reichert and A. Waibel. Online handwriting recognition: the Npen++ Recognizer. *International Journal of Document Analysis and Recognition.*2001, vol. 3, no. 3, pp. 169-180.

[4] H. Beigi, K. Nathan, G.J. Clary, and J. Subhramonia.Size normalization in unconstrained online handwriting recognition, *Proceedings ICIP.* 1994, pp. 169-173.

[5] M. Unser, A. Aldroubi, M. Eden. B-Spline signal processing: part II - efficient design and applications. *IEEE Transactions on Signal Processing.* 1993, vol. 41, no. 2, pp. 834-848.

[6] A. Sharma, R. Kumar and R. K. Sharma. Online Handwritten Gurmukhi Character Recognition Using Elastic Matching, *Congress on Signal and Image Processing.* 2008, vol. 2, pp. 391-396.

[7] E. Kavallieratou, N. Fakatakis, and G. Kolkkinakis.An unconstrained handwriting recognition system.*International Journal of Document Analysis and Recognition.* 2002, vol. 4, no. 4, pp. 226-242.

Name: Nainsi Gupta

Place of Birth: Pichhore, Shivpuri(M.P.)

Educational Background:

M.Tech (pursuing), Computer Science and Engineering, SRCEM College, Gwalior (M.P.) India

B.Tech (2009) Computer Science and Engineering, MPCT College, Gwalior (M.P.) India

Name: Mayank Gupta

Place of Birth: Mumbai.

Educational Background:

M.Tech (completed), Computer Science and Application, TU, Patiala (Punjab) India

B.Tech (2008) Computer Science and Engineering, DIT College, Dehradun (U.K.) India

Name: Rahul Agrawal

Place of Birth: Chhatarpur.

Educational Background:

M.Tech (completed), Computer Science and Application, TU, Patiala (Punjab) India

B.Tech (2009)Computer Science and Engineering, ITM College, Gwalior (M.P.) India