Online Character Recognition using Regression Techniques

Nirup Reddy, Kandan R, Shashikiran K, Suresh Sundaram, A G Ramakrishnan
MILE, Department of Electrical Engineering, IISc
Bangalore- 560012, India
nirupreddy@gmail.com, kandan.r@gmail.com, shashi.1980@yahoo.com
suresh@ee.iisc.ernet.in, ramkiag@ee.iisc.ernet.in

Abstract

This paper introduces a scheme for classification of online handwritten characters based on polynomial regression of the sampled points of the sub-strokes in a character. The segmentation is done based on the velocity profile of the written character and this requires a smoothening of the velocity profile. We propose a novel scheme for smoothening the velocity profile curve and identification of the critical points to segment the character. We also porpose another method for segmentation based on the human eye perception. We then extract two sets of features for recognition of handwritten characters. Each sub-stroke is a simple curve, a part of the character, and is represented by the distance measure of each point from the first point. This forms the first set of feature vector for each character. The second feature vector are the coeficients obtained from the B-splines fitted to the control knots obtained from the segmentation algorithm. The feature vector is fed to the SVM classifier and it indicates an efficiency of 68% using the polynomial regression technique and 74% using the spline fitting method.

1. Introduction

On-line handwriting recognition is the automatic recognition of text as it is written on a special pressure sensitive screen, where a sensor picks up the pen-tip movements X(t),Y(t) as well as pen-up/pen-down switching which is considered as a dynamic representation of handwriting. Extensive research in the past two decades has led to the development of online handwritten script recognition systems for languages like English [1], Chinese [2] and Japanese [3]. However not much attention has been given to develop similar systems appropriate to Indian languages. Symbols requiring several key strokes to define a character are a common feature of Indian languages. This attribute can be well utilized for online handwriting recognition in the Indian scenario. In this paper we attempt to evolve an online recognition system for Tamil characters. Tamil is a popular South Indian language spoken by a significant population in countries such as Singapore, Malaysia and Sri Lanka besides India. There are totally 247 letters (consonants, vowels and consonant vowel combinations) in the Tamil alphabet. Each letter is represented either as a separate symbol or as a combination of discrete symbols, which we refer to as characters in this work. Only 156 distinct symbols or characters are needed to recognize all the 247 letters in the Tamil alphabet. Samples of each of these characters constitute a separate class. So far as the work on online handwriting recognition for Tamil is concerned, Aparna et al. [4] have used string matching schemes. Dimensionality reduction techniques like Principal Component Analysis have also been employed for online character recognition. In this paper we have proposed a technique of segmenting each character at its critical points based on certain attributes of the character. After the segmentation, we obtain two sets of features and compare their efficiency. We obtain accuracy of 68% using the polynomial regression technique and 74% using the spline fit method.

2. Segmentation into Sub-strokes

There are several methods of segmenting strokes into sub-strokes such as in [5]. We have proposed two different segmentation schemes, one which reflects this variability of the velocity of motion and the other is based on the human eye perception of characters and compare it with an existing segmentation algorithm.

The first segmentation algorithm is a modified version of [6], which exploits the velocity profile of a stroke to extract sub-strokes.The trace of every stroke starts with zero velocity, accelerates and the instantaneous velocity varies as a function of the point, and decelerates to zero at the end of the stroke. If a person wants to change the direction of the trace in a stroke, it is observed that he reduces the velocity of motion and accelerates in the desired direction.

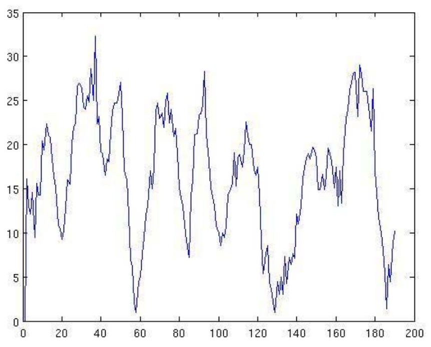

Figure 1. Velocity profile of a character

We define a sub-stroke as a part of a stroke with a change in the acceleration which reflects the intentional direction change of the stroke. Instantaneous velocity is the distance traversed by the trace between two time instants divided by the difference in the time instants. The instanteneous velocity profile is nothing but the Euclidean distance traversed between consecutive sample points as shown in equation (1).
$\forall$ t from 2 to T

$$V_U(t) = F_S\sqrt{(X(t)-X(t-1))^2+(Y(t)-Y(t-1))^2} \quad (1)$$

where $X(t)$ and $Y(t)$ are the X and Y co-ordinates of the 't_{th}' sample, T is the total number of samples in a stroke and '$V_U(t)$' is the instantaneous velocity profile. The velocity profile shows lot of variability and it is difficult to discern the segmentation points based on this kind of profile. We thus need to use a smoothening method which will remove the spurious variations in velocity. Figure 1 shows the unsmoothed velocity profile '$V_U(t)$'.

In one proposed method [7], smoothening is done with the use of two different filters, one filter which captures the local variations and the other captures the global variations. The segmentation done as per the above algorithm is shown in Figure 2 which shows the smoothened velocity profile and Figure 3 which shows the points of segmentation. However in this method the number of segmentation points for the same character may vary as per the user's average speed and sometimes this method of smoothening does not truly reflect the change in velocity at a given instant of time. This is shown in another sample of the character in Figure 4, in which the velocity profile is smoothened and Figure 5, in which the segmentation points are shown.

Hence we propose a method of smoothening of the velocity profile by averaging the velocity at every point based on the velocity of the neighbouring points. If we have a character with p number of points sampled in time then for every point k of the character, we take the velocity of n points in the neighbourhood and compute the average velocity. For the first point we calculate the average velocity as follows;

$$V_{avg} = \frac{V_1 + V_2 + ... + V_n}{n} \quad (2)$$

This is done for every sampled point in the entire character where V_k is the instantaneous velocity at every point k. Thus once the average velocity is computed as per the above algorithm, the sharper variations in velocity are much smoother now as shown in Figure 6

We now capture the defining attributes of the character based on this velocity profile by segmenting the character into a number of sub-strokes. Now to segment the character into sub-strokes, we have to obtain the critical points which reflect a change in velocity. A critical point is a point at which there is a change in the velocity as compared to a threshold and this generally happens when there is a change in the direction or a curve in the character. So in the velocity profile, we have to locate the points which indicate that there is a change in velocity or the direction of motion of the pen tip. This is done by locating the points of minimum velocity on the smoothened velocity profile. Thus a sub-stroke is obtained, when at a point t the following condition

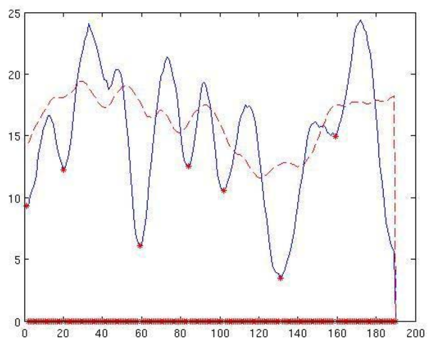

Figure 2. Smoothened velocity profile using method [7]

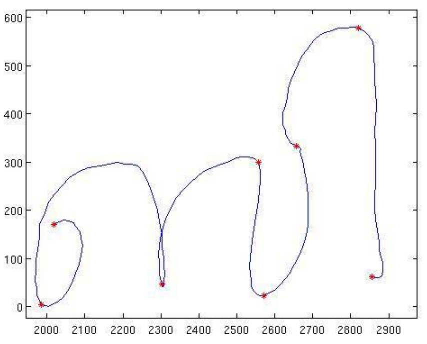

Figure 3. Points of segmentation based on [7]

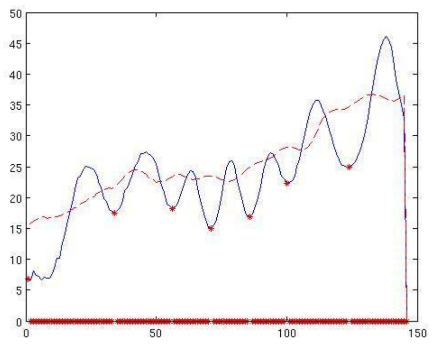

Figure 4. Smoothened velocity profile of the same character using method [7]

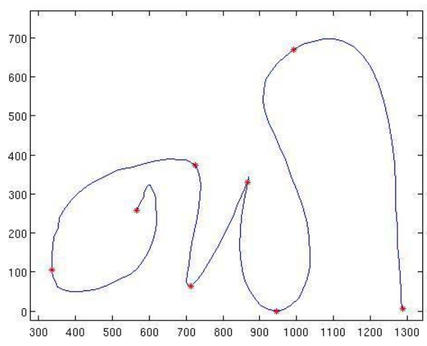

Figure 5. Variations in the number of Points of segmentation using method [7]

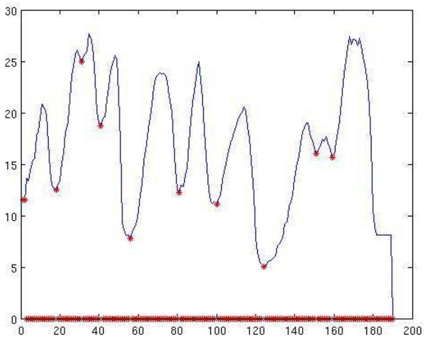

Figure 6. Smoothened Velocity profile based on velocity of neighbourhood points

is met;

$$V_{avg} = min(V(t)) \tag{3}$$

The selection of n is crucial and depends on the time sampling rate F_S of the input device and spatial resolution (dpi) of the digital screen. The segmentation points obtained are shown in 7

The above algorithm of segmentation depends on the velocity profile which has a lot of variations and hence has to be further smoothened. An alternative method is that we can locate the critical points(i.e points which indicate change in the direction of the motion of the pen tip) based on the shape of the character itself.

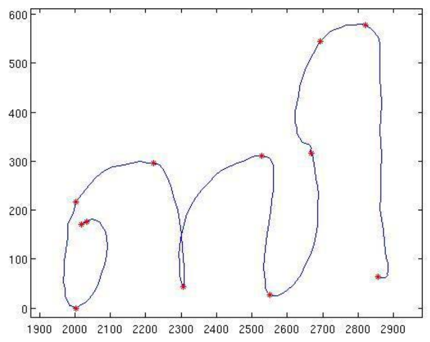

Figure 7. Points of Segmentation which indicate minimum velocity

In general the major problem in OHR is the identification of different styles of the same character as the same. Hence we propose a second segmentation algorithm to decompose each handwritten charater to its skeletal form and then identify the segmentation points. We first remove the redundancy of the points over a local region in a character and define the character with a much lesser number of points. This method is based on the perception of the human eye which can distinguish between the various characters. The human eye can account for small changes in the shape of the character and still recognize the character that is written. If we decompose the character into its skeleton, the information regarding the character is not lost but all the variablilty and the redundancy in the information is completely removed. This skeleton is still recognized by the human eye. We thus propose a method in which the character is decomposed to its skeleton and the points for segmentation is obtained from such a character by chain coding.

This is done by averaging the character over a window W_n and defining the segment of the character over the length of the window as the average value.

$$x_i = \sum_{n=1+im}^{(i+1)m} \frac{x_n}{m} \tag{4}$$

$$y_i = \sum_{n=1+im}^{(i+1)m} \frac{y_n}{m} \tag{5}$$

Here m denotes the length of the window used. We can thus see the character in its skeletal form as shown in 12.

After this, we now find the segmentation points from the

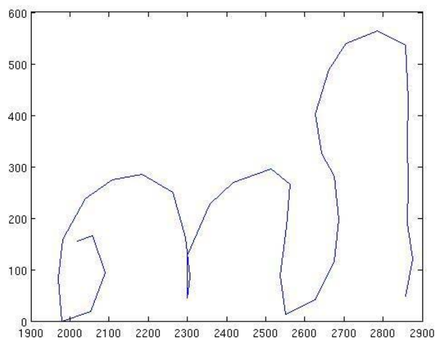
Figure 8. Skeletal form of the character

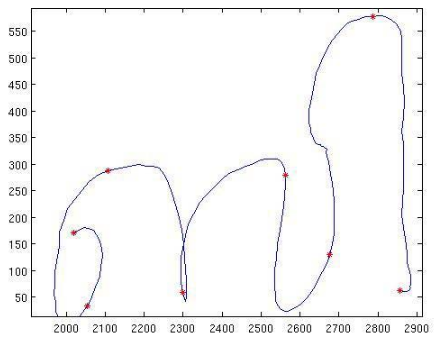
Figure 9. Points of Segmentation indicating change in direction

character in its skeletal form. This is done by the chain coding method. Let N_p be the number of points in the pre-processed character P.

- As the character is written, calculate the slope values of the sampled points of the character.
- Quantize the slope angle of the segment between two consecutive points of P into 8 levels [1]. Let Q denote the set of quantized slope values for a given P. Then we have:

$$Q = q_i| \text{ where } , i = 1, 2, 3, .., Npq_i 0, 1, 2, .., 7 \quad (6)$$

- The change in the values of the slope between the current point P_i and the slope at the previous point P_{i-1} is calculated.
- This value i.e the difference in the slope values is accumulated and when the change is greater than 180 degrees then the point is termed as the segmentation point.

This gives us the segmented parts of the character as shown in 9. Thus both the above algorithm give us accurate estimate of the critical points.

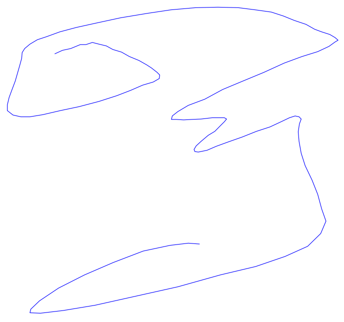
Figure 10. Sample Character written by a User

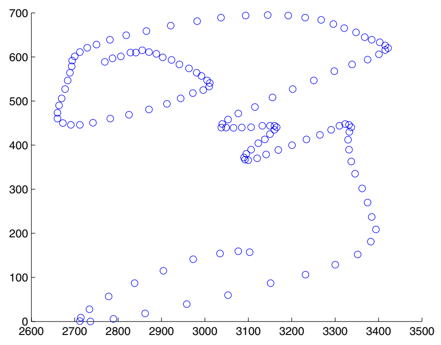
Figure 11. Character after Sampling

3. Features of the Sub-Strokes

We extract two different set of features for the character recognition. One is the polynomial coefficients of the polynomial fitted to the plot of distance and angle. Now we have to model the sub-strokes after the segmentation algorithm, which do not have any change in direction in their shape. Once we obtain the sub-stroke, we calculate the Euclidian distance of each point in the sub-stroke from the first point. So we can now plot a graph in which each point is defined as the Euclidean distance from the first point. The Euclidean distance is given by;

$$D_s = \sqrt{(X(i) - X(1))^2 + (Y(i) - Y(1))^2} \quad (7)$$

We also know that, given a discrete sampling of N data points having coordinates

$$P_i = (x_i, y_i) i = 1, 2, 3.....N \quad (8)$$

The value of y can be correlated to the value of the x coordinate via an approximate function Y having the form:

$$Y(x) = A_n x^n + A_{n-1} x^{n-1} + ... + A_1 x + Constant \quad (9)$$

which corresponds to an nth degree polynomial expansion. The expansion coefficients A_i are determined

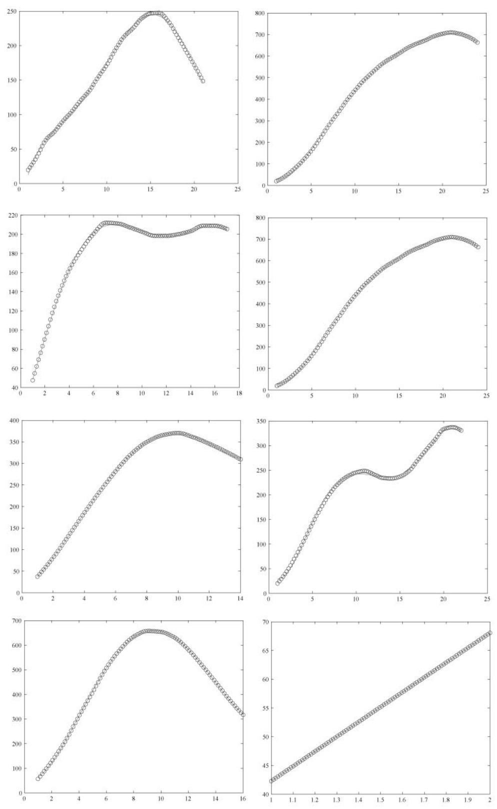
Figure 12. Polynomial fit obtained on the distances for each of the sub-strokes

by least-squares fitting the data points to this expression. The resulting continuous function may then be used to estimate the value of y over the entire x region where the approximation has been applied. We use the polynomial regresson method described above to fit a polynomial for the distances obtained for all the points in the segmented sub-stroke of the stroke. We then find the coefficients, $A_1, A_2...A_n$ of a polynomial $p(x)$ of degree n that fits the data in a least squares sense. The result P is a row vector of length $n+1$ containing the polynomial coefficients in descending powers. This is the first set of features formed for each of the segmented sub-strokes in a character. A hand written character is show in Figure 10 and the character after sampling is done is shown in Figure 11. The character shown has 9 segmentation points and thus 8 sub-strokes. Figure 12 shows the polynomial regression method applied to the distances obtained for every sub-stroke as described above.

The other set of features are the coefficients of the spline curve fitted onto the points determined by the segmentation algorithms. We use the non-uniform B-Spline curve fitting for defining the features of our dataset. A non-uniform B-

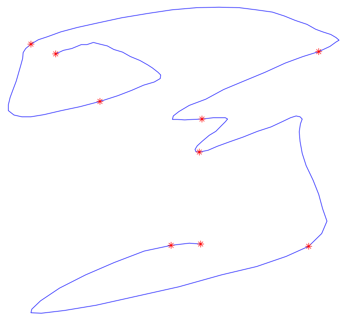
Figure 13. Critical points obtained after the segmentation algorithm

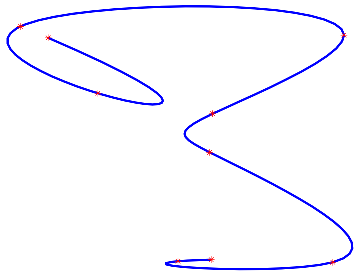
Figure 14. Spline fit with the critical points obtained from the segmentation algorithm as knots

Spline curve is defined as

$$C(t) = \sum_{i=0}^{n} N_{i,p}(t) P_i \qquad (10)$$

where P_i are the control points for the B-Spline curve and $N_{i,p}$ is the pth B-Spline basis functions defined as

$$N_{i,p} = N_{i,p-1}(t).\frac{t - t_i}{t_{i+p-1} - t_i} + N_{i+1,p-1}.\frac{t_{i+p} - t}{t_{i+p} - t_{i+1}} \qquad (11)$$

where $t_0, t_1, ..., t_{n+p}$ is a non-uniform knot vector. We choose a non-uniform B-Spline curve because it is more flexible and can reduce the fitting errors [7]. The points determined by the segmentation algorithm as the critical points are taken as the knots for the spline fitting method. Figure 13 shows the segmentation points which are used as the knots and the spline fit done on these points is shown in Figure 14.

4. Classification

We trained an SVM classifier with Radial Basis Function(RBF) kernel[8] with the polynomial coefficients and

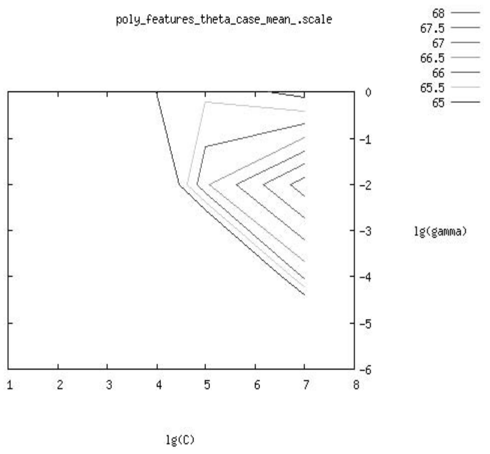

Figure 15. Five fold cross-validation results using SVM classifier

the B-spline co-efficients as the features separaely. Then SVM finds a linear separating hyperplane with the maximal margin in this higher dimensional space. C > 0 is the penalty parameter of the error term. The RBF kernel used is given below

$$K(x_i;x_j) = exp(-\gamma(||x_i - x_j||)^2, \gamma > 0 \qquad (12)$$

The kernel parameters were chosen by using a five fold cross validation using all the training data samples. The best parameters were found to be C= 128 and γ = 0.25 which was operating at an efficiency of 68% as shown in 15 for the polnomial regression technique.

5. Results and Discussion

Testing has been done on the Tamil dataset collected from 168 users with an average of 5 trials per user. The datasets has been collected using the Compaq tc 1100 tablet PC. The data collected contains characters written in various styles, number of strokes, order of strokes and direction of strokes. The results have been presented in table 1. The below results show the accuracy obtained for all the three difference segmentation algorithms described in this paper. We find that the two segmentation algorithms based on velocity profile smoothening and eye perception modelling give a much higher accuracy. Also, the spline coefficients are able to give higher accuracy for all the three types of segmentation algorithms as compared to the polynomial regression method.

6. Conclusion

We have proposed a novel writer independent recognition scheme for online Tamil characters. This is done first by the segmentation algorithms which captured the characteristics of a handwritten character. We used two different

Table 1. Comparison of Character Recognition accuracy

Segmentation Method	%Accuracy with polynomial coeff	%Accuracy with spline coeff
Existing Method	62%	67%
Algorithm based on average velocity	66%	72.4%
Algorithm based on Skeletal decomposition	68%	74%

set of features and fed it to a SVM classifier which gave us good results. The segmentation algorithms proposed by us have given very good and accurate reults and further potential research is to improve the features of the sub-strokes obtained.

References

[1] C.C.Tappert, C.Y.Suen and T. Wakahara. The state of online handwriting recognition. IEEE Trans. on Pattern. Analysis and Machine Intelligence, 12(8), 787-807, August 1990.

[2] Cheng-Lin Liu, Stefan Jaeger and Masaki Nakagawa. Recognition of Chinese Characters: The State-of-the-Art. IEEE Trans.on Pattern Analysis and Machine Intelligence, 26(2), pp.198-213, 2004.

[3] S.Jaeger, C.-L.Liu and M.Nakagawa . The state of the art in Japanese online handwriting recognition compared to techniques in western handwriting recognition . Intl Journal on Document Analysis and Recognition, Springer Berlin 6(2), October 2003.

[4] K.H. Aparna, Vidhya Subramanian, M. Kasirajan, G. Vijay Prakash, V.S. Chakravarthy, Sriganesh Madhavanath. Online Handwriting Recognition for Tamil. Proceedings of the 9th Intl Workshop on Frontiers in Handwriting Recognition (IWFHR -9), pp 438- 443 October 2004.

[5] I. Methasate and S. Sae-Tang, On-line Thai Handwriting Character Recognition Using Stroke Segmentation with HMM, Proc. Applied Informatics

[6] X. Li, M. Parizeau and R. Plamondon, Segmentation and reconstruction of Online Handwritten Scripts, Pattern Recognition, 31(6): 675-684, June 1998.

[7] T.J. Cham and R. Cipolla. Automated B-spline curve representation with MDL-based active contours. In Proc. British Machine Vision Conference, volume 2, pages 363-372, Edinburgh, September 1996.

[8] Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines, 2001. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm