# A Shape Based Post Processor for Gurmukhi OCR

G S Lehal[1], Chandan Singh[2] and Ritu Lehal[3]
[1] Department of Computer Science and Engineering, Thapar Institute of Engineering & Technology, Patiala, India.
[2] Department of Computer Science and Engineering, Punjabi University, Patiala, India.
[3] Department of Commerce(CC), Punjabi University, Patiala, India.

## Abstract

*A shape based post processing system for an OCR of Gurmukhi script has been developed. Based on the size and shape of a word, the Punjabi corpora has been split into different partitions. The statistical information of Punjabi language syllable combination, corpora look up and holistic recognition of most commonly occurring words have been combined to design the post processor. An improvement of 3% in recognition rate from 94.35% to 97.34% has been reported on machine printed images using the post processing techniques.*

## 1. Introduction

The objective of post processing is to correct errors or resolve ambiguities in OCR results by using contextual information. Dictionary look-up method [1-2] is the most commonly used post processing technique. The output of OCR is compared to system's built-in dictionary (lexicon) and candidates are generated. According to the difference between the output of OCR and the output of dictionary look-up the numbers expressing the belief in the correct classification are modified. Output sequence of suitable candidates is then ordered and the best candidate selected. Another very common post processing technique is based on statistical information about the language[3-7]. In this method, an n-gram is used to filter out unacceptable letter string candidates from the recognizer. In context of post processing techniques for Indian language script recognition systems, notable work has been done by Sinha[8] in the development of a rule based contextual post processor for Devanagri text recognition. Bansal and Sinha[9] have developed a partitioned word dictionary for correcting the optically read Devanagari character strings. The word dictionary is partitioned in order to reduce the search space besides preventing forced match to incorrect word. The word size and the envelop information of words are taken as the main partitioning features.

In this paper we describe a post processor for improving the recognition rate of an OCR of Gurmukhi script[10]. We have used a Punjabi corpus, which serves the dual purpose of providing data for statistical analysis of Punjabi language and also checking the spelling of a word. The corpus has been partitioned at two levels. At the first level the corpus is split into seven disjoint subsets based on the word length. At second level we have used the shape of the word to further segment the subset into a list of visually similar words. We have used a set of robust, font and character size independent features for identification of visually similar words. These features are available more or less as a by-product of the on-going recognition process and do not necessitate any additional computation. Holistic recognition of most commonly occurring words derived from the corpora is also carried out.

## 2. Characteristics Of Gurmukhi Script

Gurmukhi script is used primarily for the Punjabi language which is the world's 14th most widely spoken language. Some of the properties of the Gurmukhi script are:

- Gurmukhi script is cursive and the Gurmukhi script alphabet consists of 41 consonants and 12 vowels and 3 half characters, which lie at the feet of consonants.
- Most of the characters have a horizontal line at the upper part. The characters of words are connected mostly by this line called head line and so there is no vertical inter-character gap in the letters of a word and formation of merged characters is a norm rather than an aberration in Gurmukhi script
- A word in Gurmukhi script can be partitioned into three horizontal zones (Fig 1). The upper zone denotes the region above the head line, where vowels reside, while the middle zone represents the area below the head line where the consonants and some sub-parts of vowels are present. The lower zone represents the area below middle zone where some vowels and certain half characters lie in the feet of consonants. A statistical analysis of Punjabi corpus has shown the zone-wise percentage distribution of Gurmukhi symbols in printed text as : upper zone(25.39%), middle zone (70.41%) and lower zone (4.20%).

## 3. Proposed Scheme

We have used a Punjabi corpus for generating word

Proceedings of the Sixth International Conference on Document Analysis and Recognition (ICDAR'01)
0-7695-1263-1/01 $10.00 © 2001 IEEE

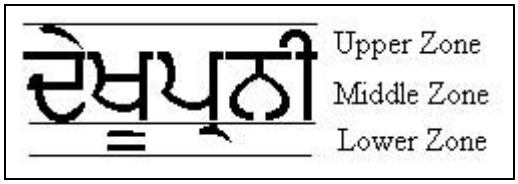


**Fig 1 : Three zones of a word in Gurmukhi script**

frequency list, which is the back bone of the post processing module. The corpus has about 0.83 million words and fifty five thousand unique words.

The main steps in the post processing phase are:

1. Create a word frequency list from the Punjabi corpus. The list stores the frequency of occurrence of all words present in the corpus.
2. Partition the word frequency list into smaller sub lists based on the word size. We have created 7 sub-lists corresponding to word sizes two, three, four, five, six, seven and greater than seven.
3. Generate from each of the sub-list an array of structures, which is based on visually similar characters. This array records the percentage frequency of occurrence of character in all the positions of visually similar words. The percentage frequency of occurrence of visually similar words is also stored in the list. This list is combined with the confidence rate of recognition of the recognizer to correct the mistakes of the recognizer.

Step 3 is explained in detail in the following sections.

## 4. Creation Of Visually Similar Word Structure List

As already discussed in previous section, the corpus is divided into seven sub-sets based on the word size. Further in each of this sub-set, a list of visually similar words is generated. We say that two words are visually similar, if each character in the corresponding position of the two words is visually similar. To decide the visual similarity of two characters, the zonal position of the character and a set of robust features is used. For this purpose, the Gurmukhi character set is divided into 16 sub-sets consisting of visually similar characters. Out of the 16 sub-sets, the first ten sub-sets(0-9) contain the characters present in the middle zone. The middle zone characters are categorized using the following four font and size invariant Boolean valued features, which have been described in detail in [10]. a) Number of junctions with the headline equals one. b)Presence of a sidebar. c)Presence of a loop excluding the headline. d)Loop formed along the headline. All the members of a sub-set share the same the Boolean values of the above mentioned features. For example, for all the members of sub-set no. 2 (Table 1), the value of the first feature is true, second feature is false, third feature is true and fourth feature is false, since all the characters in this sub-set have one branch from the headline, do not have a side bar, contain a loop but no loop is formed along the headline.

The eleventh and twelfth character sub-sets correspond to the upper zone and lower zone characters respectively. We have created separate sub-sets for some of the most frequently occurring characters, which have a very high recognition rate and are not confused with any other character. The thirteenth sub-set contains only the character ੍. From a statistical analysis of the corpus it was found that the character ੍ is the most frequently occurring character with a frequency of occurrence of 10% and it is very easily recognizable. Similarly the character ਂ , which is just a dot present in the upper zone and hereby referred as *bindi*, is very easily recognizable and not confused with any other character and has 5% frequency of occurrence, is assigned the fourteenth sub-set. The fifteenth and sixteenth character sub-sets consist of ਼ and ਼ characters. These characters, which are present in both upper and lower zones, have a high frequency of occurrence, and have no confusion with any other character. The complete sub-sets are shown in table 1.

**Table 1 : Partitioning of Gurmukhi character set into 16 sub-sets**

| Sub-set No. | Character Set | Sub-set No. | Character Set |
|---|---|---|---|
| 0 | ਚ ਰ | 8 | ੳ ੋ |
| 1 | ਹ ਜ ਜ਼ | 9 | ਏ ਝ ਵ ਲ |
| 2 | ਕ ਙ ਛ ਠ ਡ ਢ ਫ ਭ ਫ਼ | 10 | ਾ ੈ ੌ ਿ ੁ ੂ |
| 3 | ਟ ਣ ਤ ਦ ਨ ਵ ੜ | 11 | ਃ ੰ ੱ ੲ ੴ |
| 4 | ਖ ਖ਼ ਗ ਗ਼ | 12 | ੍ |
| 5 | ਥ ਬ | 13 | ਂ |
| 6 | ਅ ਘ ਪ ਮ | 14 | ਼ |
| 7 | ਸ ਧ ਯ ਸ਼ | 15 | ਼ |

As already mentioned, from the word frequency list an array of structures of visually similar words of same size is generated. We generate seven such arrays for word sizes two, three, four, five, six, seven and greater than seven. Each element in this array stores the information about the relative percentage frequency of occurrence of the visually similar words as well as the percentage frequency of occurrence of characters in different positions of the word. We call this array SSSL(Shape based Statistical Structure List). Each element of the array is assigned a unique code generated from the sub-list number of the characters and the array elements are arranged in sorted order of the code for faster searching. The structure of an element of the array SSSL is:

```
struct SSSL_element
{
  int code;
  struct char_freq_list *char_list;
  struct word_freq_list *word_list;
};
struct char_freq_list
{
   char punjabi_char;
   int frequeny;
```

```
   struct char_freq_list *next;
};

struct word_freq_list
{
   char *punjabi_word;
   int frequeny;
   struct word_freq_list *next;
};
```

As is clear from the structure, each element of SSSL has links to char_freq_list and word_freq_list. char_freq_list is a singly linked list storing the percentage frequency of occurrence of characters in a particular position and word_freq_list is a singly linked of visually similar words storing their relative percentage frequency. The header node of char_freq_list has pointers to next node of the char_freq_list and char_freq_list for next character poistion. For example, consider an eight word frequency list of words of length 3 (Table 2). From this frequency list, the SSSL with two elements as shown in Fig. 2 is generated.

**Table 2: A word frequency list of word size 3**

| Word | Frequency | Word | Frequency |
|---|---|---|---|
| ਜੋਰ | 1400 | ਹੋਰ | 2600 |
| ਹੋਰ | 500 | ਜੱਚ | 1500 |
| ਨੱਕ | 2500 | ਵੈਂਡ | 4700 |
| ਦੈਂਡ | 1600 | ਨੋਕ | 1200 |

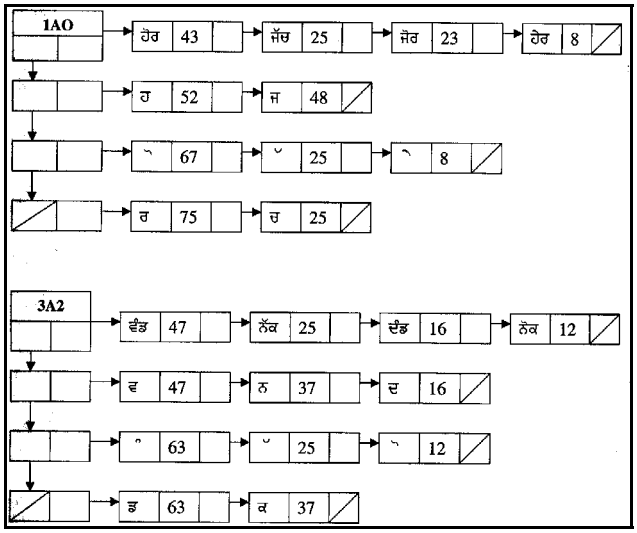


**Fig 2 : SSSL generated from word frequency list of Table2**

The first element of the list in Fig 2 contains first four words of the frequency list. These words have visually similar characters in all the three positions and hence they are considered to be visually similar. The first character in each of these words belongs to sub-set 1, second character belongs to sub-set 10 and third character belongs to sub-set zero. Thus the code of this structure in hexadecimal format is 1A0, representing the subset of the three character positions. The character ਹ is present in the first position in the first and second word, giving a total frequency 3100 and similarly the character ਜ is present in first position in the third and fourth word, giving a total frequency 2900. The percentage frequency of occurrence of ਹ and ਜ in the first position is thus 52 and 48 respectively and these values are stored in the char_freq_list for first position of the word. Similarly the percentage frequency of occurrence of characters in second and third position is calculated and the char_freq_lists for those positions are generated. Similarly the relative percentage frequency of occurrence of the words is stored in the word_freq_list.

The char_freq_lists are used to decide between confusing characters. If, for example, the recognizer determines that the first character of a three lettered word has no sidebar, one branch from the headline and no loops, the second character is present above the headline and the third character has no sidebar, one branch with the headline and one loop , then the search is made for the node having code 3A2 in the SSSL list for word size 3. From the char_freq_lists in the node one can estimate that the most probable characters occurring in the three positions are ਵ , ੰ and ਡ. This prediction will be combined with the results of the recognizer to decide the actual characters.

This recognition scheme is similar to bi-gram and tri-gram post processing analysis but it has the following advantage. In case of bi-gram analysis the prediction of the next character depends upon correct recognition of previous character. If, for example, the first character is incorrectly recognized then the bi-gram analysis will predict the second character based on the wrongly recognized first character and this result will carry on for the subsequent characters. In our current scheme, we have used very robust and font and size independent features to categorize a character to one of the sub-sets. It was observed that on clean images the characters were correctly categorized to their sub-sets in **99.82%** cases. So even if the first character is not correctly recognized, if its subset is correctly identified, the performance of the post processor is not affected. The only limitation is that, if the sub-set is not properly distinguished, then the post processor may not yield correct result. The purpose of the word_freq_list is two fold:

1. **Check for the existence of a word in the corpus:** The recognized word can be checked for its presence in the corpus. If the word is not present then it is replaced with the nearest matching word provided that the distance of the recognized characters from the stored templates is greater than some preset threshold value. This is necessary to prevent accidental conversion of non-dictionary words such as proper nouns and abbreviations to a dictionary word.
2. **Perform holistic recognition of a word:** The words in the list are sorted in descending order of frequency of occurrence. If it is found that the first word in the list has frequency of occurrence greater than 90, then the recognized word is converted to the high probability word again subject to the condition that

the distance of the recognized characters from the stored templates is greater than some threshold value. It was also observed from the word frequency list that the twenty most commonly occurring words in the corpora occupy 20% of the Punjabi text. So any visually similar word to these common words is automatically converted to one of these words.

## 5. Characteristics Of SSSL

As already stated, the Punjabi corpora is used to generate frequency list of all the words in the corpora and from this list SSSL for words of sizes two, three, four, five, six, seven and greater than seven are generated. Table 3 depicts the characteristics of SSSL of different sizes. The first column represents the size of a word in the SSSL. The number of unique words of a particular size are shown in second column, while the third column depicts the number of elements in the SSSL. The maximum and average number of nodes of word_freq_list are shown in following columns. The second table stores the relative percentage frequency of number of words in word_freq_list. It is worth noting that a smaller count of words in word_freq_list leads to holistic recognition of a word. If for example, there is only one word in the word_freq_list, then any other visually similar word can safely be recognized as that lone word in the list.

**Table 3 : Characteristics of SSSL of different sizes**

| SSSL Word Size | Total words | No. of elements in SSSL | Max. nodes | Average nodes |
|---|---|---|---|---|
| 2 | 81 | 45 | 6 | 1.8 |
| 3 | 4782 | 1088 | 60 | 4.40 |
| 4 | 12480 | 4920 | 49 | 2.54 |
| 5 | 14200 | 9069 | 22 | 1.57 |
| 6 | 10994 | 8938 | 11 | 1.23 |
| 7 | 7763 | 6444 | 18 | 1.20 |
| >7 | 9199 | 8455 | 7 | 1.09 |

| SSSL Word Size | Relative % of occurrence of words in word_freq_list (No. of nodes in word_freq_list) | | | | |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | >4 |
| 2 | 55.56% | 24.44% | 11.11% | 4.44% | 4.44% |
| 3 | 27.94% | 21.50% | 12.22% | 8.27% | 30.06% |
| 4 | 50.44% | 20.65% | 10.35% | 6.18% | 12.38% |
| 5 | 69.80% | 17.92% | 6.45% | 2.60% | 3.23% |
| 6 | 83.79% | 12.12% | 2.45% | 1.06% | 0.57% |
| 7 | 87.45% | 9.23% | 1.78% | 0.54% | 0.99% |
| >7 | 92.63% | 6.31% | 0.67% | 0.30% | 0.07% |

## 6. Application Of The SSSL In Post Processing

The recognizer generates a character distance pair set (CD){(c1, d1), (c2, d2)}, where c1 represents the best matching character and d1 represents the distance of the input character image from the character prototype stored in the training data. Similarly c2 and d2 represent the second nearest matching character and the distance of the input character image with the character prototype. The CD pairs are stored for each character position in the word. The SSSL is combined with the CD pairs to predict the most likely character. The final decision of the choice of character is obtained by combining the results of the recognizer and the post processor. Depending on how closely the character image matches with the nearest character prototype and the second nearest character prototype, a decision is made on how much weightage has to be assigned to the post processor and the recognizer. We have used two weights w1 and w2, where w1 is based on the distance of the character with its nearest matching prototype(d1) and represents the confidence of the recognizer. Smaller value of w1 means that the character image is closely matching the library image and lesser weight has to be assigned to the post processor. The weight w2 represents the closeness of the shapes of the top two choices. Higher value of w2 means that the shape of the character images of top two choices are closely matching and the confusion is to be resolved by assigning more weight to the post processor. There is a quadratic growth of the weights with the distance. These weights are combined with the percentage of occurrence of the top two choices at a particular position in the word and the distance of top two choices with their nearest matching training set prototypes. The decision on the choice of one of the characters is taken as follows:

We calculate a parameter, dist, which represents the distance of the recognized character from the actual character. This has been formulated empirically as

$$dist = ((w1+w2)/(100.0))*(p2-p1) - (d2-d1)^2 \quad (1)$$

where d1 = distance of first choice (char1) with the nearest matching library prototype
d2 = distance of second choice (char2) with the nearest matching library prototype
$w1 = d1^2$ subject to maximum of 50;
$w2 = 50-(d2 - d1)^2$ subject to minimum of 0;
p1 = percentage frequency of occurrence of char1 in char_freq_list;
p2 = percentage frequency of occurrence of char2 in char_freq_list;

We recognize a character as char2 if dist > 0 else it is retained as char1

### 6.1 Some Samples

Consider the following skeletonized images .

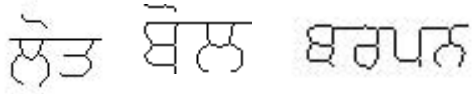

**Fig 3 : Sample images**

The first image in Fig 3 represents the word ਲੋੜ but it has been identified as ਲੋਤ by the recognizer, since the limbs

of ੜ have been removed during the binarization and thinning stage. The CD pair generated for the third character is {(ੜ ,2 ) , (ੜ, 6)}. The percentage frequency of occurrence of ੜ and ੜ in the third position in the corresponding char_freq_list is found to be 0 and 90 respectively. The calculated values are d1=2, d2=6, p1=0, p2=90, w1=4, w2=34 and dist=18 and since dist is positive so ੜ is replaced by ੜ

Using this technique we have been able to rectify more than one wrongly recognized character in a word. The second image in Fig 3 represents the word ਬੋਲ but it has been identified as ਥੋਲ by the recognizer. The CD pairs generated for the first two wrongly identified characters of the word are {(ਥ ,1 ) , (ਬ,2)} and {( ੋ,4 ) , ( ੋ,6)}. The percentage frequency of occurrence of ਥ and ਬ in the first position in the corresponding char_freq_list is 0 and 100 respectively. Thus d1=1,d2=2, p1=0, p2=100, w1=1, w2=49 and dist=49. For second character d1=4, d2=6, p1=1 and p2=99, w1=16 and w2=46 and dist=56.Since dist is positive for both the cases so the second choice character is taken for both cases and thus the word is corrected as ਬੋਲ

The third image in Fig 3 represents the word ਬਚਪਨ but it has been identified as ਥਰਪਨ by the recognizer. The CD pairs generated for the two wrongly identified characters are {(ਥ ,3 ) , (ਬ,5)} and {( ਰ ,2 ) , ( ਚ,6)}. The percentage frequency of occurrence of ਥ and ਬ in the first position in the corresponding char_freq_list is found to be 0 and 100 respectively. Similarily the percentage frequency of occurrence of ਰ and ਚ in the char_freq_list for second position is 0 and 100 respectively. Thus dist is found to have positive values for first and second character positions in the word and so the best choices are replaced by the second best choices and the word is correctly identified as ਬਚਪਨ

## 7. Experimental Results

We have tested the performance of the OCR and the post processor on about forty text images which included scanned images from books, newspapers and laser print outs. The recognition accuracy of the OCR without post processing was 94.35%, which was increased to 97.34% on applying the post processor to the recognized text. Punjabi grammar rules have also been used to check for illegal character combinations. The recognition rate for the characters in all the three zones is tabulated in table 4.

**Table 4 : Zonal wise Recognition accuracy of the OCR with and without the application of post processor**

| Zone | Recognition rate without post processing | Recognition rate after post processing |
|---|---|---|
| Upper zone | 91.19% | 95.14% |
| Middle zone | 96.71% | 98.38% |
| Lower zone | 79.48% | 87.87% |

Table 5 lists some of the most difficult to recognize characters for the OCR whose recognition accuracy was substantially increased by the application of the post processor. These characters were difficult to recognize because of the confusion with other similar characters (ਚ confused with ਰ), (ਬ confused with ਥ), ( ੁ confused with ੂ and ੍ ) and ( ੈ confused with ੇ and ੋ ).

**Table 5 : Some of the most difficult characters to recognize**

| Character | Recognition rate without post processing | Recognition rate after post processing |
|---|---|---|
| ਚ | 70.56% | 91.41% |
| ਬ | 77.78% | 95.72% |
| ੈ | 78.24% | 85.65% |
| ੁ | 65.90% | 81.46% |

## References


1. C.J. Wells, L.J. Evett, P.E. Whitby and R.J. Whitrow,"Fast dictionary lookup for contextual word recognition", *Pattern Recognition*, 23(5), pp. 501-508 (1990).
2. E. Mayes, F.J. Dameran and R.L. Mercer, "Context based spelling correction", *Information Processing and management*, 27(5), pp. 517-522 (1991).
3. X.Tong & D.A. Evans, "A statistical approach to automatic OCR error correction in context", *Proceedings of the 4th workshop on very large corpora*, pp. 88-100 (1996).
4. C. Y. Suen, "N-gram statistics for natural language understanding and text processing", *IEEE Transactions on Pattern analysis and Machine Intelligence*, 1(2), pp. 164-172 (1979).
5. E.J. Yannakoudakis, I. Tsomokos and P.J. Hutton, "N-grams and their implication to natural language understanding", *Pattern Recognition*, 23(5), pp. 509-528 (1990).
6. E. M. Riseman and A. R. Hanson, "A contextual postprocessing system for error correction using binary n-grams", *IEEE Transactions on Computers*, c-23(5) , pp. 480-493 (1974).
7. J. J. Hull and S. N. Srihari, "Experiments in text recognition with binary n-gram and Viterbi algorithm", *IEEE Transactions on Pattern analysis and Machine Intelligence*, *4*(5), pp. 520-530 (1982).
8. R.M.K. Sinha, "Rule based contextual post-processing for Devanagri text recognition", *Pattern Recognition*, *20*(5), pp. 475-485 (1987).
9. V Bansal and R.M.K. Sinha, "Partitioning and searching dictionary for correction of optically read Devnagri character strings", in *Proceedings International Conference on Document Analysis and Recognition*, pp.653-656 (1999).
10. G S Lehal and Chandan Singh, "A Gurmukhi script recognition system", in *Proceedings 15th International Conference on Pattern Recognition*, Vol 2, pp. 557-560 (2000).