Evaluation of Feature Sets in the Post Processing of Handwritten Pitman's Shorthand

Swe Myo Htwe, Colin Higgins
The University of Nottingham,
School of Computer Science and IT,
Wallaton Road, Nottingham, NG8 1BB, UK
smh@cs.nott.ac.uk, cah@cs.nott.ac.uk

Graham Leedham, Ma Yang
Nanyang Technological University,
School of Computer Engineering,
N4-#2a-32 Nanyang Avenue, Singapore 639798
asgleedham@ntu.edu.sg, mayang@pmail.ntu.edu.sg

Abstract

Innovative ways to rapidly input text becomes essential in today's world of mobile computing. The paper discusses the computer transcription of handwritten Pitman shorthand as a means of rapid text entry to pen-based computers, particularly from the aspect of linguistic post processing. Feature-to-phoneme conversion is introduced as the first stage of a text-interpreter and the application of various production rules based on different pattern structures is discussed. It demonstrates that phoneme ordering is compulsory in dictionary-based transcription and the use of an approximate pattern-matching algorithm resolves the problem of recognition confusion between similar patterns. Experimental results are promising and demonstrate an overall accuracy of 84%.

1. Introduction

Handheld computing creates an environment where people have both mobility and the ability to send, receive and process information. Whilst today's handheld devices have transformed into powerful pocket-sized computers, the transformation of a standard "QWERTY" keyboard into these handheld devices has not been so effective. Miniature keyboards make text entry very slow (less than 10 words per minute (wpm) [1]. Handwriting recognition systems like Unistroke and Graffiti are alternative means of text input to pen-based computers, but writing individual characters on a smooth digitizer still results in slow text input of less than 10wpm. Other rapid handwriting input methods like, T-Cube [2] and Quickwriting [3], allow a user to compose entire words or even sentences as a single outline. However their restrictions towards gestures (*e.g. a user must never stop moving the stylus until a word or a sentence has been fully written*) is not conducive to a natural feeling of writing. High-speed text entry is particularly essential for mobile rapid note-takers. Today's stenographers spend time transcribing paper-based shorthand notes because their handheld devices like Tablet PCs or Personal Digital Assistants (PDAs) are not productive enough to record speech in real time. It is therefore appropriate to develop a technique in which text can be written on a digitizer in a very comfortable and natural way, preferably at the speed of speech (120 – 180 wpm).

This paper proposes the computer transcription of Pitman shorthand as a means to increase the compatibility of handheld devices in the real-time reporting industry. With a Pitman shorthand recognizer, users could input text at an average rate of over 100 words per minute by using standard shorthand notations and semantic transcription can be achieved by the use of menus, approximate phoneme matching, or an automatic collocation analyzer. In this paper, we overview the overall process of the recognition and interpretation of handwritten Pitman shorthand and then introduce an efficient algorithm by which basic primitives of Pitman shorthand such as loops, circles, strokes or hooks are interpreted into orthographic English.

1.1. Background

Pitman shorthand was invented by Sir Issac Pitman in 1873. There are two main forms: - classic New Era and New Pitman 2000. Although both of them are based on the same principles, New Era notation is slightly faster to write but more difficult to learn than the Pitman 2000. It was widely used in offices in the UK and also taught in 74 other countries [4]. Pitman's shorthand records speech phonetically and it comprises phonemes of 24 constants, 12 vowels, and 4 diphthongs. It also defines

Proceedings of the 9th Int'l Workshop on Frontiers in Handwriting Recognition (IWFHR-9 2004)
0-7695-2187-8/04 $20.00 © 2004 IEEE

IEEE COMPUTER SOCIETY

approximately 90 of the most frequently used English words as short-forms i.e., special signs written without vowel components. Basic notations of Pitman shorthand and sample outlines are illustrated in *Figure 1*.

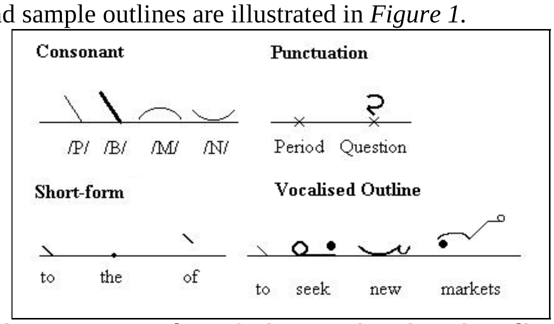


Figure 1. Samples of Pitman shorthand outlines

The potential of Pitman shorthand [5][6] as a means of rapid pen-driven text entry to a computer has been reported since the 1980's. Subsequent works by Brooks et al [7][8] introduced the computer transcription of Pitman's shorthand as an aid for the deaf by providing a real-time transcript of lectures and meetings. Research in the 1990's[9][10] classified the whole Pitman shorthand into two types: *vocalized outlines and shortforms* and applied different AI methods in their recognition and transcription. Another recent approach by Nagabhushan et al [11][12] concentrated on the post processing of segmented basic features such as loops or hooks being interpreted to English text using heuristic methods.

2. Overview of Our System

Shorthand outlines are fed into the recognition engine as shown in *Figure 2* and differentiated between a vocalized outline and a short-form. Short-forms are recognized separately from vocalized outlines using a Template Matching Algorithm in which a ranked list of English words is produced. Therefore, the transcription system no longer needs to be concerned with short-forms at the word level transcription. As for vocalized outlines, the recognizer segments them into basic primitives i.e., strokes, hooks, circles or loops using dominant point information as a threshold value. Then, the segmented data are matched up with the features of basic consonants and categorized into a ranked list of phoneme primitives using a neural network classifier. Due to the difficulty of accurately detecting pen pressure on normal digitizers, phonemes with different line thickness like "P" and "B" are clustered as the same type. In some cases, the classifier is unable to classify a single primitive as a whole consonant outline, for example, outlines of "W" and "Y" are classified separately as a small hook (or) and a stroke written upwards . Therefore, an additional step is needed to correctly interpret clustered primitives into related phonemes. We refer to this process as *"Feature to Phoneme Conversion (FtoP)"*. In an early part of *"FtoP"*, we use an approximate pattern-matching algorithm in which user-dependent stroke variations or wrong vowel allocations are adjusted. The output of "FtoP" is a ranked list of phonemes and they are later discriminated by a phonetic dictionary and matched to a list of homophones at the word level transcription. In sentence level transcription, the homophones are passed through a collocation analyzer, where each word is justified by its frequent use with another word or phrase and the most probable word is selected by the system i.e., "*Internet*" in the following example (*Figure 2*).

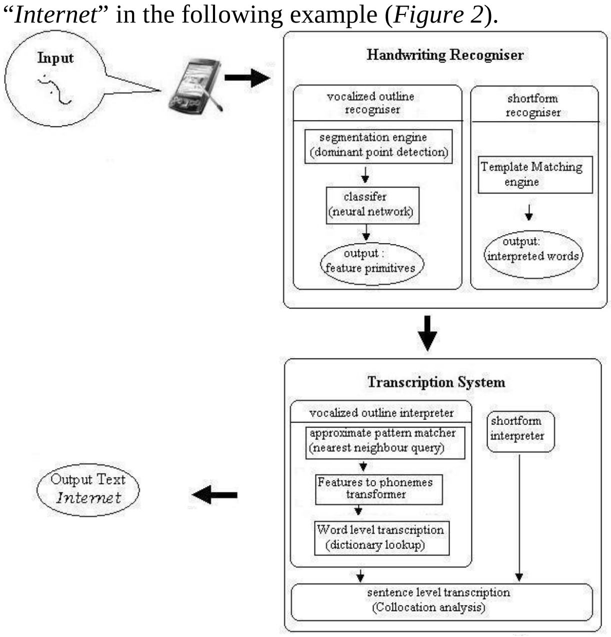


Figure 2. Illustration of the computer transcription of Pitman shorthand

3. Transcription

3.1. Approximate pattern matching

Handwritten outlines are bound to differ in angular structures from writer to writer and even from the same writer from time to time. Another problem in the transcription of handwritten Pitman shorthand is that vowels are not written clearly between dot and dash and neither are they put at an accurate position *i.e., at the beginning, middle or end of an outline*. In the pre-processing, the recognition engine detects a ranked list of candidates for individual primitives, but it cannot cope with user-dependent or accidental pen strokes. For example, if the "T" consonant is accidentally written as a vertical curve instead of a vertical straight stroke, the recognizer does not estimate the curve as a vertical stroke.

IEEE COMPUTER SOCIETY

Approximate pattern matching is, in fact, a heuristic approach in which uncertain pen strokes are coped with and wrong vowel locations are estimated. It uses the nearest neighbor query and the heuristic is based on the similarity between the two patterns.

Function Name **:** Return value
APPROXIMATE_PATTERN_MATCH(pattern) **:**
A list of similar primitives
Begin
  pattern: a sample input pattern
  Heu-Fn : a heuristic function
  Return **NEAREST-NEIGHBOR-QUERY**(pattern, Heu-Fn)
End

The heuristic function (Heu-Fn) can be defined as

$H_{neighbours}(p)$ = nearest neighbor primitives which are similar to pattern 'p'

Primitives with similar structure are defined as the nearest neighbors and the whole Pitman primitives are located to seven neighborhoods where, four of them are related to consonant kernels, one to circular primitives and the remaining two to vowel primitives. Here, *similarity* stands for, "having similar angular structure" for consonant kernels, "having similar shape" for circular components and "having similar location and shape" for vowel primitives. Samples of neighborhoods are shown in *Figure 3* and the use of approximate pattern matching algorithm is illustrated in *Figure 4*.

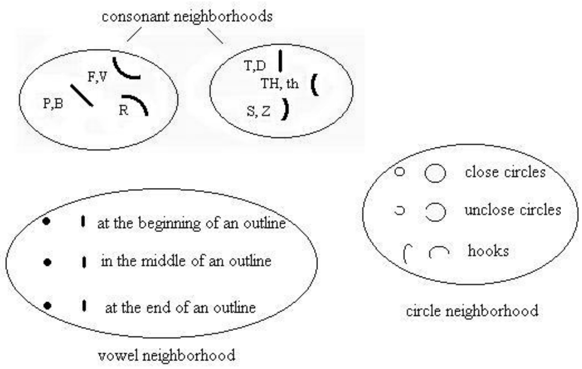


Figure 3. Samples of neighborhood used in approximate pattern matching algorithm

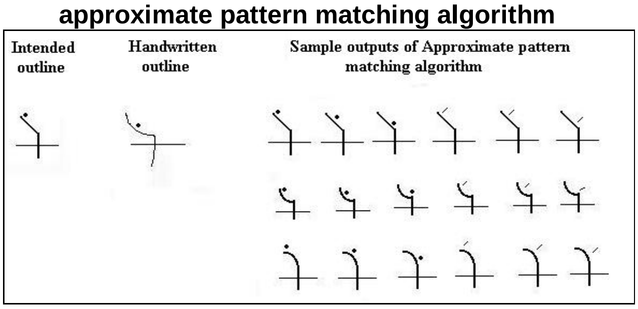


Figure 4. Approximate pattern- matching using nearest neighbor query

3.2 Features to Phonemes Conversion

Once a vocalized outline has been classified into pattern primitives of hooks, circles and strokes, it is necessary to convert these feature primitives into a phonetic representation. In fact, around 20% of the pattern primitives can be directly mapped to basic consonants, and the remaining 80% of the primitives need to be translated by the application of production rules of Pitman shorthand. Similar to the work of Leedham and Downton [4], the production rules are applied with respect to the stroke primitives of the consonant kernel, and their adjacent primitives.

Basically, there are five production rules introduced in our system: - "D*irect Translation (DT)", "Primitive Combination (PC)", "Primitive Combination and Reverse Ordering (PCRO)", "Feature Detection (FD)" and "Length Detection (LD).* Phonemes related to each rule are as follow:

- *Direct translation rule:* all consonants except Y, W and H
- *Primitive combination rule:* W, Y, H
- *Primitive combination and reverse ordering rule:* PL, BR, etc, PR, BR, etc., FR, VR, etc., and FL, VL etc
- *Feature Detection rule:* SES, ZES circles, ST, STER loop, N, F, V, SHUN hook, suffix –SHIP hook, suffix –ING/INGS dot
- *Length Detection rule:* MD, ND, suffix –MENT, half length strokes, double length strokes

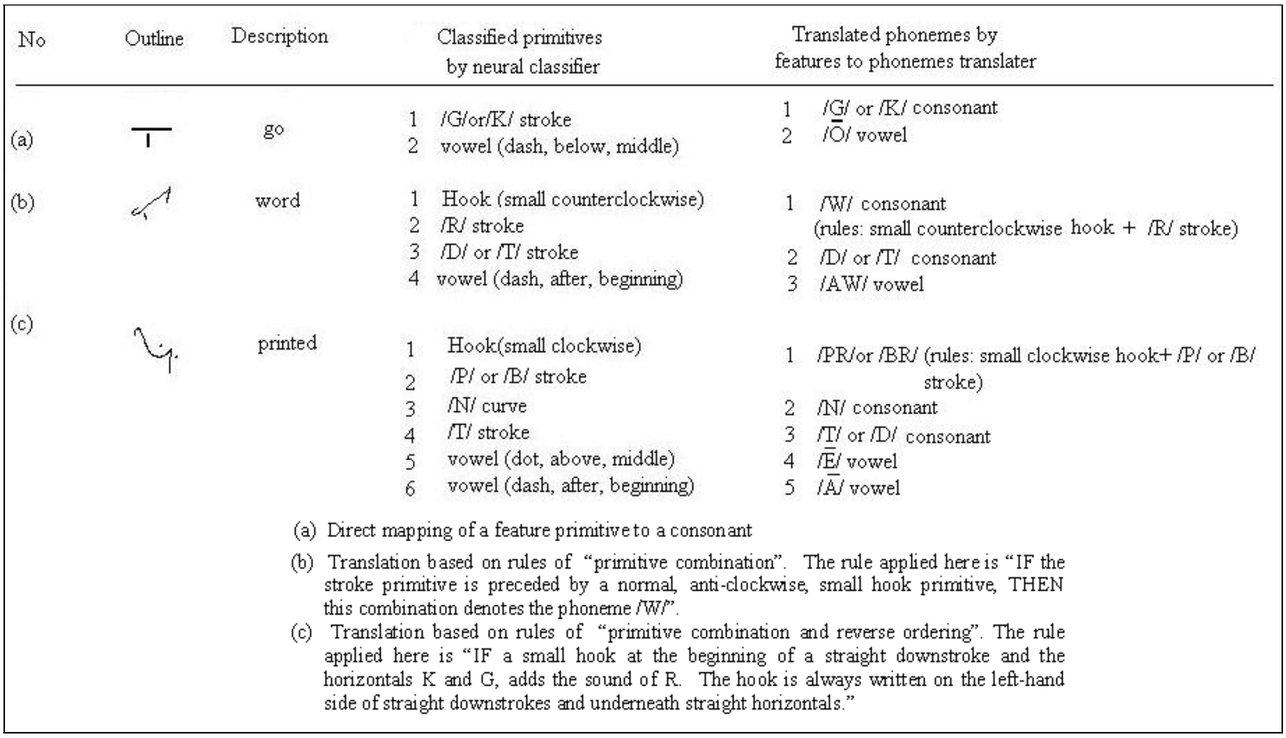
To clarify the first three rules, consider the three sample outlines in *Figure 5* and to clarify the last two rules, refer the three examples below.

Example 1: Application of Feature Detection (FD) Rule

Circle SES: Pitman uses a large circle to indicate the sound of SES, SEZ, ZES or ZEZ at the end of an outline. For this case, one of the FD rules is read: - "*IF the stroke or curve primitive is followed by a large circular loop primitive, THEN the loop appends /SES/, /SEZ/, /ZES/ and /ZEZ/ to the preceding phoneme."*

Example 2: Application of Length Detection (LD) Rule

Half-length stroke N, M: For the purpose of speed writing, Pitman uses a halved and thickened N or M stroke or just halves the length of the N stroke to indicate the succeeding sound of D or the suffix /MENT/ respectively. For this case, one of LD rules is read: - "*IF M or N curve is halved in length, THEN the half-length curve inserts /D/ and /MENT/ after the phoneme of /N/ or*



No	Outline	Description	Classified primitives by neural classifier	Translated phonemes by features to phonemes translater
(a)		go	1 /G/or/K/ stroke 2 vowel (dash, below, middle)	1 /G/ or /K/ consonant 2 /Ō/ vowel
(b)		word	1 Hook (small counterclockwise) 2 /R/ stroke 3 /D/ or /T/ stroke 4 vowel (dash, after, beginning)	1 /W/ consonant (rules: small counterclockwise hook + /R/ stroke) 2 /D/ or /T/ consonant 3 /AW/ vowel
(c)		printed	1 Hook(small clockwise) 2 /P/ or /B/ stroke 3 /N/ curve 4 /T/ stroke 5 vowel (dot, above, middle) 6 vowel (dash, after, beginning)	1 /PR/or /BR/ (rules: small clockwise hook+ /P/ or /B/ stroke) 2 /N/ consonant 3 /T/ or /D/ consonant 4 /Ē/ vowel 5 /Ā/ vowel

(a) Direct mapping of a feature primitive to a consonant

(b) Translation based on rules of "primitive combination". The rule applied here is "IF the stroke primitive is preceded by a normal, anti-clockwise, small hook primitive, THEN this combination denotes the phoneme /W/".

(c) Translation based on rules of "primitive combination and reverse ordering". The rule applied here is "IF a small hook at the beginning of a straight downstroke and the horizontals K and G, adds the sound of R. The hook is always written on the left-hand side of straight downstrokes and underneath straight horizontals."

Figure 5. Phonemes translation using DT, PC or PCRO rules

*/M/."*Examples of such an outline is shown in *Figure 6*. In fact, length is not actually detected by the recognizer system and a normal stroke N and a halved stroke N are classified as the same primitive. In order to avoid confusion between full and half strokes, the transcription system detects dominant points of N and M curves and evaluates their length. If it is detected as a half stroke, the system appends the consonant /D/ and /MENT/ to any halved N and M stroke. For example, the phonetics of "madam" transcribed by the system are /**M** Ĕ **D** Ă M/ or /**M** Ĕ **M** Ĕ **N** **T** Ă M/. Obviously, the latter phoneme set does not exist in English language and it will later be removed by a phonetic dictionary.

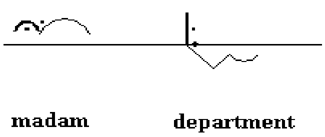


Figure 6. Samples of half-length /M/ stroke and half length /N/ stroke

<u>Example 3: Application of Length Detection (LD) Rule</u>

Double length strokes: All curved strokes are doubled in length to present the addition of the syllables –TER, -DER, -THER and –TURE in Pitman shorthand. For this case, another LD rule is read: - *"IF a curve primitive is doubled in length, THEN the double-length curve inserts /TER/, /DER/, /THER/ and /TURE/ after the phoneme of the curve. Any vowel assigned to the curve should be read before the insertion."* Similar to the half-length translation algorithm, the system detects the length using dominant point information. If it is detected as a double-length, the outline will firstly be taken as normal length and be translated into related phonemes using the "Direct

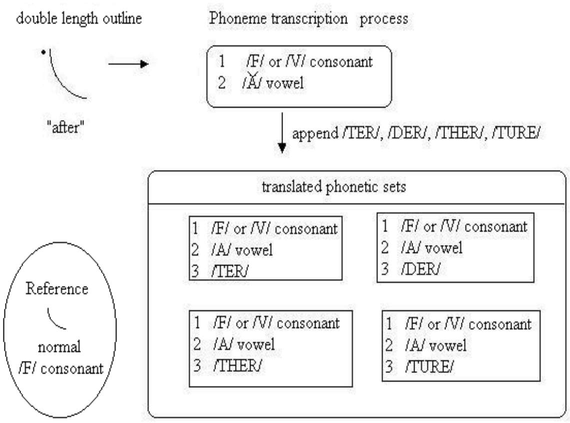


Figure 7. Sample of phoneme translation of a double length stroke

0-7695-2187-8/04 $20.00 © 2004 **IEEE**

```
Initial order of primitives in a vocalized outline "Writing"

                 WStart                 (Description: word start)
1st Primitive    S1,0,64,13,0.093493    (Description: consonant No, start coordinate, end coordinate, consonant type, probability)
2nd Primitive    S2,64,137,2,0.931558
3rd Primitive    S3,19,-1,24,1
4th Primitive    V1,0,64,2,1,3          (Description: vowel No, start coordinate, end coordinate, sequence, position, type)
                 Wend                   (Description: word end)

Sorted list

                 WStart                 (Description: word start)
1st Primitive    S1,0,64,13,0.093493    (Description: consonant No, start coordinate, end coordinate, consonant type, probability)
2nd Primitive    V1,0,64,2,1,3          (Description: vowel No, start coordinate, end coordinate, sequence, position, type)
3rd Primitive    S2,64,137,2,0.931558
4th Primitive    S3,19,-1,24,1
                 Wend                   (Description: word end)
```

Figure 8. Phoneme sorting based on vowel insertion

Translation" principle, but at the end of processing, phonemes of /TER/, /DER/, /THER/ and /TURE/ will be inserted next to the phonemes of the double length curve. To illustrate this principle, consider the example in *Figure 7*.

3.3 Phoneme ordering

The reason phonemes need to be sorted after the conversion of features to phonemes is because pattern primitives are not classified in linguistic order at the recognition stage. The recognition first detects the whole outline and produces a list of consonant primitives in sequential order. Vowels are then recognized from left to right order and appended at the end of the consonant primitives. Therefore, the phoneme ordering requires correct insertion of vowels among the consonant kernels. The beauty of the recognition engine is it can not only detect an approximate dominant point of a vowel primitive, but also can estimate the possible sequence of this primitive against nearby consonants (*i.e., whether the vowel occurs before or after the consonant outline)*. The system uses "*Dominant Point based Insertion (DPI) algorithm*" for phoneme ordering in which vowels are assigned to related consonants using dominant point information and inserted at the right order using the sequence information. The algorithm is illustrated by an example in *Figure 8*. In this example, as the vowel "V1" falls in the same coordinate range as the consonant "S1", it is moved to the adjacent place of S1 and the sequence identifier (*"2" in this case*) indicates that it is pronounced after the consonant "S1".

4. Evaluation

The current goal of our experiment is to evaluate the overall performance of the recognition and transcription engines. With the help of three experienced shorthand writers, a sample sentence consisting of 28 vocalized outlines and 20 short-forms was entered to our recognition system nine times *(3 times by each writer*). The sample sentence used in this experiment was chosen to reflect the general domain area and the words contained in the most frequently used 5000 English words provided at "edit" *http://www.edict.com.hk/textanalyser/*. It was also chosen to cover the most likely pen-stroke combinations in Pitman shorthand. The input outlines were segmented and classified into pattern primitives by our neural network classifier and then interpreted to orthographic target scripts. Input and output pairs of the experiment are shown in *Table 1* and they are evaluated from the following three main aspects: - *accuracy of final text output; consequence of inconsistent writing styles from writers to writers; and the impact of approximate pattern matching algorithm in the recovery of classification errors*.

Table 1. Comparison of input shorthand-outlines with interpreted outputs

No	Written outline	Interpreted words	No	Written outline	Interpreted words	No	Written outline	Interpreted words
1	writing	-	17	more	more	33	have	have
2	and	and	18	expensive	expensive	34	turned	-
3	books	books	19	to	to	35	out	out
4	survived	-	20	produce	-	36	to	to
5	even	-	21	for	for	37	be	be
6	after	after	22	the	the	38	faster	faster
7	the	the	23	same	same	39	and	and
8	invention	invention	24	quantity	quantity	40	easier	easier
9	of	of	25	of	of	41	to	to
10	film	film	26	ideas	ideas	42	reproduce	-
11	radio	radio	27	only	-	43	and	and
12	and	and	28	computer	-	44	spread	spread
13	television	television	29	storage	storage	45	than	than
14	which	which	30	and	and	48	the	the
15	are	are	31	the	the	47	printed	-
16	much	much	32	Internet	-	48	Word	word

IEEE COMPUTER SOCIETY

Experimental results show that 84% of the written outlines can be interpreted correctly, but the remaining 16% failed to produce related words. In addition, a very interesting phenomenon observed in the experiment is 37% of perfect transcription occurs in the presence of recognition errors. This proves that the approximate pattern-matching algorithm is capable of dealing with recognition errors between circles and loops, curves and strokes, etc.

Another important factor observed in this experiment is the omission of vowels in an outline causes a total failure of transcription. This is a critical concern for further research, as it is a common practice of stenographers to omit vowels in an outline depending on their experience or individual inclination.

Another useful finding of this experiment is the system hits a complete failure when the input outlines are legible to human readers, but are not exactly consistent with the writing rules of Pitman shorthand. Although the natural feeling of writing is a primary concern of our handwritten recognition research, disagreement with writing rules is not allowed in our system.

5. Conclusion

The experimental results are promising, but further works need to be done in both the recognition and transcription engines. Firstly, the recognition system needs to produce more accurate feature sets. In the current experiment, feature sets are sometimes miss-classified and it imposes "*Nearest Neighborhood Function* " to consider wider neighborhoods and makes the search exponential. Another improvement proposed in the recognition stage is to detect vowels' location more accurately with respect to a consonant kernel, as 50% of the current recognition error comprises wrong attachment of vowels to nearby consonants. The solution to the problem of vowel omission should be followed up in the near future and the use of a modified phonetic dictionary without vowel components is expected to be a promising approach. However, the anticipated downside of this approach is there will be a high proportion of homophones for each outline and the final word selection imposes a closer look up on collocation analysis (*statistics of a word or phrase which is frequently used with another word or phrase*). Therefore, the next goal of our research is to produce an algorithm for the sentence level transcription of handwritten Pitman shorthand outlines, taking homophones as a primary problem to resolve.

Acknowledgement

We would particularly like to thank members of Nanyang Technological University for arranging a workshop and contributing to the success of the experiment.

Reference

[1] Toshiyuki M., 'POBox: An efficient text input method for handheld and ubiquitous computers', Proc. of the ACM Conference on Human Factors in Computing System (CHI'98), Los Angeles, USA, pp. 328-335, April 1998,

[2] Venolia D., and Neiberg F. 'T-Cube: A fast, self-disclosing pen-based alphabet.', Proc. of ACM Conference on Human Factors in Computing System (CHI 94), Addison-Wesley, pp. 265-270, April 1994

[3] Perlin K., 'Quick writing: Continuous stylus-based text entry', Proc. Of the ACM Symposium on User Interface Software and Technology (UIST'98), Santafe, NM, pp. 215-216. November 1998

[4] Leedham C.G., Downton A.C., 'Automatic recognition of Transcription of Pitman's Handwritten shorthand', In Plamondon R. and Leedham C.G. (Eds), Computer Processing of Handwriting, pp.235-269, World Scientific, 1990

[5] Leedham C.G., Downton A.C., Brooks C.P. and Newwell A.F., 'On-line acquisition of Pitman's handwritten shorthand as a means of rapid data entry', Proc. 1st Int. Conf. On Human Computer Interaction, London, UK, pp. 2.86-2.91, 4th-7th Sept. 1984

[6] Leedham C.G. and Downton A.C., 'On-line recognition of Pitman's shorthand: an evaluation of potential', Int. J. Man-Machine Studies, Vol.24, pp.375-393, 1986

[7] Brooks C.P., 'Computer Transcription of Pitman's Handwritten shorthand for the dead', PhD Thesis, Department of Electronics, University of Southampton, 1985.

[8] C.P. Brooks, A.F. Newell, 'Computer transcription of handwritten shorthand as an aid for the deaf - a feasibility study', International Journal of Man Machine Studies, 1985, vol. 23, pp.45-60

[9] Y. Qiao and C.G. Leedham, 'Segmentation and recognition of handwritten Pitman shorthand outlines using an interactive heuristic search', Pattern Recognition, 1993, vol.26, No.3, pp.433-441

[10] M. Zhu, Z. Chi, and X.P. Wang, 'Segmentation and recognition of on-line Pitman shorthand outlines using neural network', Proc. International Conference on Neural Information Processing, Singapore, 18-22 Nov. 2002, vol. 5, pp.2454-2458.

[11] P.Nagabhushan and Basavaraj.Anami, 'A knowledge-based approach for recognition of handwritten Pitman shorthand language strokes', Sadhana, Journal of Indian Academy of Sciences, Vol. 27, Part 5, pp. 685-698, December 2002

[12] P.Nagabhushan and Basavaraj.Anami, 'Dictionary Supported Generation of English Text from Pitman Shorthand Scripted Phonetic Text', Language engineering conference, Hyderabad, India, pp.33 December 13-15, 2002