Syntax Directed On-Line Recognition of Cursive Writing

Yung Taek Kim

Introduction. Two dimensional input devices of the computing machines attract many new studies in the man-machine communication field.

Although a two dimensional computer input is very attractive, a limited pattern recognition capability surely restricts its practical value.

For this reason, those works directed toward high dimensional interactive computer studies in software as well as in hardware have been discouraged.

Various experiments have been performed in recognition of hand written, single characters and symbols on two-dimensional computer input devices during the past decade (2, 3). Experiments were also conducted in the recognition of connected words of handwirting, using, dictionary-driven word matching programs for limited combinations of characters (2, 5).

In contrast, the object of the experiment reported here is to recognize arbitrary, handwritten connected words as well as single character. In order to achieve this result, a different approach was required.

For the recognition of cursive writings without dictionaries or large tables, the writing was cut at the middle point of each down cave of the writing, and a syntax was organized for these strokes using a hierarchical system which is determined by the characteristics of all characters.

A sementics is implemented for each level of hierarch to evaluate the characteristics for more efficient discrimination at each lebel the hierarchy.

Stroke Naming and the Relative Position Coding. Each stroke was broken into several branches to code the direction of the branches for naming of the stroke.

As soon as the stroke changes its major direction, a new directional code is

assigned, which is added to the previous code multiplied by ten; this is repeated until the coding of the entire stroke is completed.

Four diagonal major directions are used in this work, and they are defined below:

if X and Y both decrase then Code=0;
if X decreases and Y increases then Code=1;
if X increases and Y decreases then Code=2;
if X and Y both increase then Code=3;

The horizontal stroke and short stroke were specially coded using 7 in this work.

In this procedure the stroke coding has less than six or seven digits because normal hand writing stroke does not have more than five or six directional changes.

In the program, the positions were named as following:

if normal sized then pocode=1 else
if above positioned then pocode=2 else
if down positioned then pocode=3 else
if full sized then pocode=4 else then pocode=0;

Syntax for Cursive Writing Recognition. The group of strokes for each level is selected by the stroke characteristics and stroke functions. The selection is made such that a processing at a given level is indepenent of any lower level process that may exist.

The highest level is processed first using its own strong characteristics with highest reliability for recognition of the writing. Then. the next higher level is processed in the same manner using its own characteristics with a high reliability for the recognition and without further reference to any higher levels.

The other low level would follow the above routine for their individual processing steps until all the strokes were processed.

The Baye's theorem (8) can analyze this organization of writing recognition in terms of conditional probability. As long as the higher level keeps better reliability, this syntactic organization would promise optimal reliability for the

entire experiment.

In Table I the syntax specifications are shown using a list processing language in a formalism similar to Bakus normal form (1).

The first classification, second classification, third classification, fourth classification, fifth classification, sixth classification, would form a complete routine for writing recognition in case of the users who write well.

A correcting loop which includes the third and fourth classification is inserted between the second classification and the fourth classification of the above system to correct in third classification and fourth classification of the system.

A correcting loop is shown in the syntax specifications of Table I, and the iteration idea is also shown in the entries of the specifications. The semantics is explaind in the next section.

TABLE 1

Syntax specification for handwriting recognition with self correcting loop

⟨characters⟩	:=⟨first classification⟩ ⟨second classification⟩ ⟨correcting loop⟩ ⟨fifth classification⟩ ⟨sixth classification⟩
⟨correcting loop⟩	:=⟨o-correction⟩/⟨b-correction⟩
⟨0-correction⟩	:=⟨third classification⟩/⟨first classificarion⟩ ⟨second classification⟩
⟨b-correction⟩	:=⟨fourth classification⟩⟨first classification⟩ ⟨second classification⟩ ⟨third classification⟩
⟨first classification⟩	:=⟨unique character⟩/⟨stairway character⟩
⟨second classification⟩	:=⟨intersecting character⟩/⟨pointed character⟩
⟨third classification⟩	:=⟨circled character⟩
⟨fourth classification⟩	:=⟨first-fixed character⟩
⟨fifth classication⟩	:=⟨osculated character⟩/⟨last-fixed character⟩
⟨sixth classification⟩	:=⟨relative character⟩

⟨unique character⟩	:='z'
⟨stairway character⟩	:='y'/'k'/'s'
⟨intersecting character⟩	:='x'/'k'/'t'
⟨pointed character⟩	:='i'/'j'
⟨circled character⟩	:='a'/'d'/'g'/'o'/'q'
⟨first-fixed character⟩	:='b'/'c'/'l'/'f'/'h'/'k'/'v'/'w'
⟨osculated character⟩	:='m'/'n'
⟨last-fixed character⟩	:='v'/'w'
⟨relative character⟩	:='c'/'e'/'p'/'r'/'s'/'u'/'v'/'w'/'y'

From Table II, the character 's' or can be first side of any character having the children node as the next side of the character depending on the neighboring conditions.

The following list shows the summarized program for the priority in stroke combination and evaluation of the characteristics of each stroke for the tree of the stroke 30-1 as shown in Figure 1.

```
if SD(I)=30-1
    and SD(I+1)=30-1
        check 's' calling SCHEK
        if true then CHA(I)='s'
        if false then check 'v' at children nodes calling VCHEK
          if true CHA(I)='w'
          if false then CHA(I)='u'
    and SD(I+1)=30-3
        check 's' calling SCHEK
        if ture then CHA(I)='s'
        if false then CHA(I)='y'
    and SD(I+1)=302-1 or 32-1
        check 's' calling SCHEK
        if true then CHA(I)='s'
        if false then check 'v' calling VCHEK
```

if true then CHA(I)='v'

if false then check 'v' at childrein nodes calling VCHEK

if true then CHA(I)='w'

if false then CHA(I)='n'

The feedback theory for the stability of the software iterating system has been implemented in this experiment during the organization of the syntax and the criteria of the stability for itertive system is discussed in this section.

After the procedures of figure extraction, the program will start to recognize the stroke combination as the character composition by checking the errors in character recognition, and it would correct the rest of the information if any error is found by the syntax analyzer.

The corrected information requires another iteration to check and possible error and take new stroke combination for the recognition.

	30-1	30-3	302-1	3102-1	32-1	320-1	3202-1
	s		c e r s	ce	r	s r	r s
30-1	u w	p	u w		u w		
30-3	y		y			y	y
302-1	u v w	p	u v q		u v w		v
3102-1							
32-1	u v w		u v w				v
320-1		p					
3202-1		p					

TABLE II. Stroke combinations for relative character

The number of iterations is decided by the error correcting algorithm.

To design a general system the stability of the hardware system may be checked by the Routh Cirteria (9).

For the self-corrective system by iteration, the stability will be designed by the syntax organization, and the priority of the characteristics has to be studied

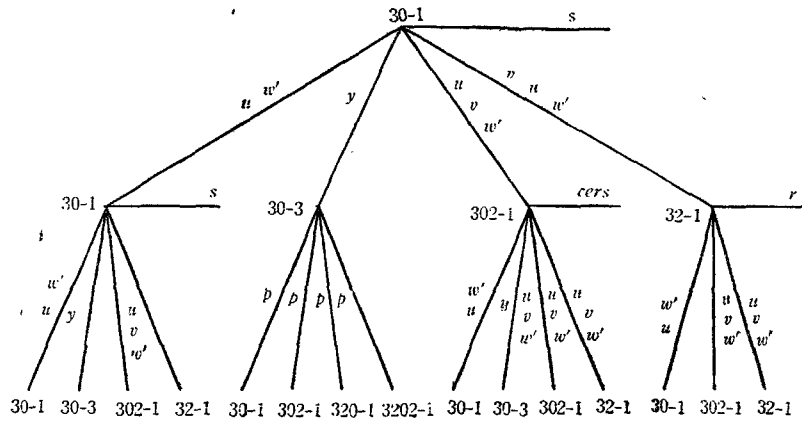

Figure 1. Tree organization for the relative stroke 30-1

for each stroke, and the level of error has to be qualified by the types of errors. The hierarchical organization for such work is always complicated, but a good solution gives always a reliable recognition.

Self-Correction by Iteration. The syntax organization for this system has dealt with the error corrections of cutting algorithms and the error corrections in stroke combination for character compositions.

The major types of error in this experiment are classified as the decision error in relative discriminating routines.

The regular writing would match the bottom of each character to the base line of the writing except the strokes of positon code 3 and position code 4.

The iterative algorithm for this position coding routine cover the wider range of writings which do not match the base line of the writing.

Conclusions. The syntax-directed algorithm for handwriting recognition was constructed using self-correcting loops to cheek for writing errors and to correct the errors made by normal writers. A simple experiment without feedback loops for hand-writing recognition was constructed during the early period of laboratory work and the recognition was highly reliable for writers who had short trainining using the standard styles.

For more reliable recognition for uncareful writers, the feedback loops were implemented to eliminate the major failures in recognition by the first system.

The recognition rate for the system with feed-back loops was quite relable and the types of error which were the major errors in the early system were eliminated, and the variations in allowable writings for normal users were much broader.

To improve the recognition for all types of writing, a number of feedback loops should be implemented. Such increased capability to recognize all types of writing may be of limited value because of increased cost of operation. The feedback loops were implemented in such a way as to widen the range of users and also keep the efficiency of the program high.

Bibliography

1. Backus, J.W., *Revised Report on the Algorithm Language Algol* 60, Assn. Computing Machinery Communications, Vol. 6, No. 1, pp. 1-17, Jan. 1963.
2. Eden, M., *Handwriting and Pattern Recognition,* IRF Trans. on Information Theory, Vol. IT-8, pp. 160-166, 1962.
3. Groner, G.F., *Real-Time Recognition of Hand Printed Text,* Am. Federation of Information Processing Soc. (AFIPS)—Fall Joint Computer Conference Proc., pp. 591-601, 1966.
4. Highleyman, W.H., *Linear Decision Functions, with Application to Pattern Recognition,* IRE Proc., Vol. 50, pp. 1501-1514, 1962.
5. Mermelstein, P., and M. Eyden, *A System for Automatic Recognition of Handwritten Words,* Am. Federation of Information Processing Soc. (AFIPS)—Fall Joint Computer Conference Proc., Vol. 26. pp. 333-342, 1964.
6. Nagy, G., and G.L. Shelton, Jr., *Self-Corrective Character Recognition System,* IEEE Trans, Vol. IT-12. No. 2, pp. 215-222, 1966.
7. *Noble*'s *Better Handwriting for Everyone,* Noble and Noble Publishers Inc., New York, N.Y., 1962.
8. Papoulis, A., *Probability, Random Variables and Stochastic Processes,* McGraw-Hill Book Company Inc., New York, N.Y., 1965.

Seoul National University