CVPR-86

A COMPUTATIONAL APPROACH TO VISUAL WORD RECOGNITION: HYPOTHESIS GENERATION AND TESTING

Jonathan J. Hull and Sargur N. Srihari

State University of New York at Buffalo
Department of Computer Science
Buffalo, New York 14260

ABSTRACT

An algorithm for reading images of words of text is presented and analyzed. The algorithm is based on a model of the human reading process which suggests that word recognition has two stages: hypothesis generation and hypothesis testing. Given an input word, hypothesis generation finds a group of candidate words from a given vocabulary. Hypothesis testing is a feature testing strategy to discover the word in this group that best matches the word in the input image. This paper concentrates on the hypothesis testing stage, which is formulated as a tree search problem in which a small number of tests are executed to recognize an input word. A statistical study using a text of over 800,000 words shows that at most three tests, from among 212 different tests, would have to be executed to recognize any word in that text. A complete analysis of this method is also given for smaller texts. Experiments with word images that show the feasibility of this technique are also described, e.g., images of 100 words in five different fonts were recognized with 92% to 97% accuracy.

1. INTRODUCTION

The reading of text by computer is a topic that has been investigated for many years; surveys include [14] and [8]. Most solutions to this problem rely on a process whereby an input word is segmented into isolated characters and these characters are exhaustively recognized. However, this approach is probably not robust enough to handle the wide range of fonts and scripts that people can easily read. Recent research in this area has recognized the importance of word level contextual information to the recognition process[4, 11]. However, these methods have been used only after characters have been isolated and recognized. The work presented here seeks to bypass the segmentation phase and apply word level information directly to the recognition stage.

This paper discusses part of a methodology for solving the machine reading problem that seeks to achieve human competence in recognizing text. This methodology seeks to adapt portions of what is known about the human reading process to the development of robust machine reading algorithms. This type of approach shares some similarity to the three levels of understanding discussed by Marr in [7]. A similar line of inquiry was followed by Shillman for isolated character recognition [10]. Shillman determined the parameters of a character recognition process using psychological experiments. This led to an algorithm that, in some cases, was able to out-perform humans [12]. The interaction of knowledge about human reading and computational techniques was also observed by Brady [1] who speculated on the importance of this interaction for the development of a better understanding of both human reading and its computational realization.

The first and most obvious observation about human reading is that it is not a character by character recognition process. This explanation was rejected late in the nineteenth century when it was discovered that people could recognize a four letter word in about the same amount of time it took to recognize a single character[2]. Some current explanations of human word recognition include at least two steps: hypothesis generation by the wholistic analysis of word images, followed by a hypothesis testing process that uses information from the hypothesis generation phase to carry out a detailed analysis of the input[9, 13].

The hypothesis generation and hypothesis testing components of human reading are adopted here as the components of an algorithm for reading text. Hypothesis generation uses a small number of features extracted from the image of an input word to locate a subset of words in a given vocabulary that should contain the input word. Hypothesis testing uses the subset of words returned by the hypothesis generation stage to construct a search space where the nodes represent tests that discriminate between n different features. There are n outgoing arcs from each node, each of which represent the presence of one of the n features. Each node is also associated with a set of words. At the root, this set of words contains the result of the hypothesis generation stage. There is no feature test associated with the root. A descendent of a node is associated with a smaller set of words than its immediate ancestor. A terminal node has no feature test but has a set of one word associated with it. Therefore, any path from the root to a terminal node represents the application of a sequence of tests to an input word. The results of the tests are indicated by the branch taken at each node in the path. The word associated with the terminal node is the one recognized in the input.

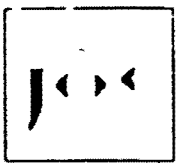
The following sections of this paper discuss the algorithms for hypothesis generation and hypothesis testing. The application domain is restricted to lower-case printed text. The primary concern of this paper is hypothesis testing. Hypothesis generation was discussed in [5] and is summarized in the next section.

2. HYPOTHESIS GENERATION

The hypothesis generation component of the algorithm uses a description of the gross visual characteristics of a word image to index into a dictionary and retrieve a subset of words called a *neighborhood* that have the same description. The visual description is the left-to-right sequence of occurrence of a small number of features. The features are simple and easy to extract to increase the reliability of the technique in the presence of noise. This type of description was used in [3] in a one-step method for cursive script word recognition. However, this approach is better for generating hypotheses about an

Appeared in the Proceedings of IEEE-CS Conference on Computer Vision and Pattern Recognition, Miami Beach, FL June 1986, 156-161.

(a)



(b)

comp. type	min.x	min.y	max.x	max.y	size
dot	26	151	51	176	519
vert	28	35	47	114	1287
vert	61	77	76	107	323
vert	108	78	123	106	291
vert	140	85	156	114	347

(c)

051110

(d)

Figure 1. Example of the image processing operations used in hypothesis generation. (a). shows an input word in the Bodoni Bold font, (b). shows the vertical bars detected in (a). (c) is the symbolic description of the components in (b), and (d) is the shape number derived from (c).

input word, as is done here, since a small number of features can serve to partition a large dictionary into a limited number of small subsets [5]. This is less error-prone than using many features to carry out complete recognition. The features used to compute the visual description of a word image are shown below:

0. A significant filled space at the beginning or end of a word (e.g., the space to the right of the vertical part in the 'c');
1. A short vertical part (e.g., the leg of an 'r');
2. A long vertical part extending above the main part of the word (e.g., the ascender portion of a 'b');
3. A long vertical part extending below the main body of the word (e.g., the descender in a 'p');
4. Dots over short vertical parts (occurs in an 'i');
5. Dots over long vertical parts (occurs in a 'j').

The visual description for each character is shown below:

a	01	h	21	o	11	u	11
b	21	i	4	p	31	v	0
c	10	j	05	q	13	w	0
d	12	k	20	r	10	x	0
e	10	l	2	s	0	y	0
f	20	m	111	t	1	z	0
g	11	n	11				

The visual description of a word is represented by appending the visual descriptions of its characters. Any zero that appears between two non-zeroes is deleted and at most one zero is retained at the beginning or end. For example, the visual description of "dog" is "121111", the visual description of cat is "111", and the visual description of "tie" is "1410". The zeroes are deleted because it is easier to detect filled spaces at the ends of words than in the middle. This representation is then used to partition a dictionary of words into subsets that all have the same visual description.

This representation is able to produce small subsets in large vocabularies [5]. For a vocabulary of 43,264 words, 28% of the vocabulary was uniquely specified by the representation. Also, the average size of a partition of the vocabulary was only 2.5. It is interesting to note that this vocabulary represents over 1,000,000 words of running text.

The representation is also demonstrably reliable. An algorithm was able to correctly compute the shape numbers in 85% to 100% of 1500 word images. The remaining cases were all rejected, i.e., there was 0% error rate. Figure 1. illustrates the image processing operations used by this program.

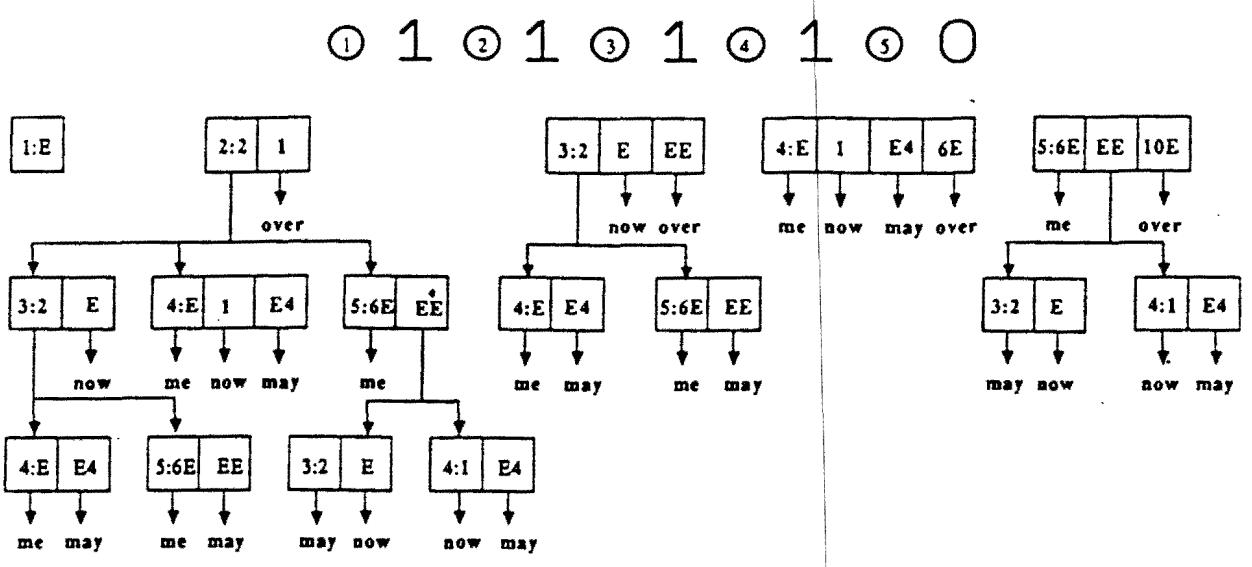


Figure 2. An example search space for the neighborhood { me, now, may, over } with shape number "11110". Each node contains a position indicator "n:" where n is one of 1 through 5. The test at each node is carried out by discriminating between the features that follow the position indicator.

3. HYPOTHESIS TESTING

The hypothesis testing strategy presented here is given a neighborhood and uses the words in that neighborhood to determine a feature testing sequence. This testing sequence is adaptable to differences in low-level visual characteristics, such as different fonts. This low-level adaptability is reflected in a change in testing sequence when it is discovered that the visual characteristics of the input have changed.

Another characteristic of the hypothesis testing strategy is the ability to alter its feature testing strategy based on the high level goal of the reading process. This is carried out in human reading when people are "reading for content" as opposed to when they are "reading for pleasure". Different feature testing strategies are used in both cases. The high level adaptability is desirable in a reading algorithm in similar situations. For example, if an algorithm is reading instructions about the dosage of prescription medicines, a very precise technique is required. However, a degree of imperfection might be acceptable if an algorithm was reading a newspaper article.

3.1 Hypothesis Testing as Search

The desirable characteristics of a feature testing strategy outlined above are realized by a search control structure. This control structure determines the order in which features are tested in a word image and uses the results of those tests to determine subsequent tests. The result of each test reduces the number of words that could match the input image. Low-level adaptability is provided by the interpretation that takes place during the search. Unreliable results in a situation where reliable results were previously attained causes backtracking to take place. High level adaptability is provided by allowing different levels of redundancy in the testing process. A rigorous strategy of reading for content requires more confirming evidence during the search than does the less rigorous one of reading for pleasure.

The search strategy is best explained by first recalling that in the hypothesis generation process, every word in a neighborhood is characterized by the same visual description. The left-to-right sequence of digits in the visual description corresponds to the left-to-right sequence of features in the input word.

Individual words in the neighborhood are distinguished by what occurs *between* the features in their visual description. This characteristic is used in the search by first quantifying these *inner-features* and then using a testing strategy that, at each step in the search, discriminates among the limited number of inner features that can occur in a specific position. The inner-features used here are:

feature code	description
E	empty space;
1	closed at both the top and bottom, e.g., 'o';
2	closed at the top, e.g., 'n';
3	closed at the bottom, e.g., 'u';
4	left of a short vertical part in an 'a';
5	right of a short vertical part in a 'c';
6	right of the short vertical part in 'e';
7	right of a long vertical part in an 'f';
8	between two short vertical parts in a 'g';
9	right of a long vertical part in a 'k';
10	right of a short vertical part in an 'r'.

An example is now used to explain the hypothesis testing strategy. Figure 2 shows the hypothesis testing search space for the neighborhood { me, now, may, over } that has the visual description "11110". The areas between each vertical part are numbered from left-to-right with the digits 1 through 5, as shown at the top of the figure. The first level of the tree shows the different possibilities for tests in each of the five

vocab. size	words in text	% voc. uniq	no. of ngl	max. ngl	avg. ngl	no. ambig. trees	max.shrtst path	avg.shrtst path	avg. avg. path
10	246,640	100	0	0	0	0	0	0	0
50	413,565	70	6	3	2.5	0	2	1.2	1.4
100	483,355	55	16	4	2.8	1	2	1.1	1.5
250	566,288	51	39	8	3.1	2	2	1.1	1.5
500	632,693	49	74	10	3.5	3	2	1.2	1.6
750	674,112	47	109	12	3.6	3	2	1.3	1.6
1000	705,045	46	147	14	3.7	5	2	1.3	1.6
2000	781,581	45	283	24	3.9	9	3	1.3	1.6
3000	827,178	42	428	32	4.0	8	3	1.3	1.7

Table 1. Characteristics of the search trees of the given vocabularies. Notation: "voc. uniq" is the percentage of the vocabulary uniquely specified by the shape measure; "ngl" is a neighborhood that contains greater than one word, thus, e.g., "avg. ngl" is read as the average size of the neighborhoods that contains greater than one word.

positions. The first position can only contain an empty space. This does not reduce the solution space. The second position can contain either feature 1 (closed at the top and bottom), which occurs between the two short vertical parts in an 'o', or the second position can contain feature 2 (closed at the top only), which occurs between the first two short vertical parts of an 'm' or an 'n'. If feature 1 is found, the input is recognized as the word "over". If feature 2 is found, the solution space is reduced to { me, now, may } and the search continues. If neither feature 1 nor feature 2 is found in the second position, the input is rejected.

Continuing the example, if feature 2 is found in position two, positions three, four and five are considered next. If position three is empty, "now" is recognized, and if position three contains feature 2, the solution space is reduced to { me, may } and either positions four or five are tested to carry out the recognition. A similar process would take place along the other paths in the tree. The difference between each path is the tests at the nodes. These tests can vary in difficulty. For example, { me, now, may, over } may be completely recognized by a single test in position four. This is a four-place test, as opposed to the sequence of a three-place test followed by a two-place test that would be needed if the search started in position three.

Low-level adaptability is provided by associating a confidence measure with the result of each test. High level adaptability is provided by altering the decision process. If the algorithm is "reading for pleasure", the first successful path is accepted. If the algorithm is "reading for content", more than one path is explored. Each final decision must agree for the correct word to be output. Otherwise, a ranked set of decisions is output.

3.2 Statistical Study of Hypothesis Testing

Various characteristics of the search trees are interesting if this methodology is to be used for reading large vocabularies. These characteristics include the overall number of trees needed for a given vocabulary. This is the same as the number of neighborhoods that contain more than one word. The average number of words in these neighborhoods indicates the size of the initial solution space for a typical tree. Since it is also possible that the given inner-feature set is unable to discriminate a given set of words, the number of ambiguous search trees is also of interest.

An interesting aspect of the search trees are the shortest paths from the root to a terminal node in each tree. The number of nodes on such a path is equivalent to the number of tests that would have to be executed to recognize any word in the neighborhood that corresponds to that tree. If the maximum length of the shortest path in all the trees of a given vocabulary is small, then the maximum number of tests that would have to be executed to recognize any word in that vocabulary is also small. The average shortest path in all neighborhoods is also of interest since it indicates the average number of tests required by any tree. The average of the average path length in all the trees indicates the number of tests needed to explore the average tree. This shows the number of tests needed by an adaptive testing strategy that could begin at any position.

The statistical characteristics of search trees were computed for vocabularies of from 10 to 3000 words. The results of this study are presented in Table 1. The vocabularies were extracted from the Brown corpus [6] which is a 1,013,549 word text designed to be representative of contemporary American English. This text was converted to all lower case and the individual words were broken out and sorted by their frequency of occurrence in the corpus. The top n most frequent words were used to construct a vocabulary of size n, where n ranged from 10 to 3000, as shown in the first column of Table 1. The second column of Table 1 shows the number of words in the corpus that are represented by the vocabulary size in the first column. The percentage of the vocabulary that is uniquely specified by its visual description is shown in the third column of the table. This is the percentage of the vocabulary that has a neighborhood that contains only itself. The other vocabulary words all fall in a neighborhood of greater than one word. The number of neighborhoods that contain more than one word is given in the column labeled "no. of ngl". The maximum and average number of words in neighborhoods that contain more than one word are shown in the columns headed "max. ngl" and "avg. ngl". The number of neighborhoods where every word cannot be uniquely recognized is shown in the column labeled "no. ambig. trees". The maximum and average length of a shortest path in every tree are given in the columns labeled "max. shrtst path" and "avg. shrtst path". The overall average of the average length path in every tree is given in the column labeled "avg. avg. path".

Some interesting results of this study include the average number of words in a neighborhood of size greater than one, the number of ambiguous trees, as well as the characteristics of the shortest paths in each tree. The average number of words in a neighborhood of more than one word reaches only 4.0 for a vocabulary of 3000 words, which accounts for more than 83% of the entire corpus. Thus, the average solution space contains only 4.0 words. The number of ambiguous trees only reaches 8 out of 428 totals trees, or about 2%.

The highest average shortest path for any vocabulary is only 1.3 for the 3000 word vocabulary. This says that on the average, only 1.3 tests are needed to recognize a vocabulary of 3000 words. Similarly, the average length of all paths in all the search spaces ranges from 1.4 to 1.7. This is an even more encouraging result which states that the average neighborhood needs only this many tests to discriminate between its words.

The characteristics of the tests at the nodes in the search trees were also studied. The number of different tests in all of the trees tells the number of tests that must be programmed if an exhaustive testing strategy is used. The number of different tests on a single shortest path chosen from each neighborhood indicates the minimum number of tests needed. These figures are shown in Table 2. It is interesting that the number of different tests in all parts of all the trees only goes up to 923. Also, the minimum number of tests needed if a single shortest path is chosen from each neighborhood ranges up to 212 for the 3000 word vocabulary. The first shortest path in each neighborhood was chosen for the purpose of this study. A procedure is currently being developed that will find the minimum number of tests where each test is ranked according to difficulty. It is expected that an even smaller number of tests will be found with this procedure.

vocab. size	different tests in all trees	different tests on one shortest path in each tree
10	0	0
50	13	6
100	35	14
250	91	30
500	199	59
750	282	80
1000	350	101
2000	631	151
3000	923	212

Table 2. Characteristics of the tests in the search space. The total number of different tests in all parts of all the trees is shown, along with the number of different tests on a shortest path chosen from every search space.

4. EXPERIMENTAL RESULTS

The viability of the hypothesis testing strategy and its ability to adapt to different input conditions was demonstrated by its implementation for a vocabulary of 100 words. These are the 100 most frequent words in the Brown corpus that were part of the statistical studies discussed earlier. Each of these words was generated in five different fonts. The fonts were 24 pt. samples of Americana, Baskerville Bold, Bembo, Bodoni Bold, and Bodoni Bookface. These font samples were digitized at 500 pixels per inch binary on a laser drum scanner. Examples of each of the fonts are shown in Figure 3. The word images were generated by appending the images of the appropriate characters. This resulted in 500 input images.

The "reading for pleasure" testing strategy was used for this experiment. This strategy utilizes the tests on a shortest path in each search space. For the purpose of this experiment, the first shortest path in each of the 16 search spaces was used. This required the programming of only the 14 different discrimination tests shown below:

1.	(1 E)	8.	(1 3)
2.	(6E E)	9.	(6E 10E)
3.	(6E 10EE EE)	10.	(1 3 E)
4.	(E EE)	11.	(6E 6EE)
5.	(6EE 10E EE)	12.	(6E E EE)
6.	(1 2 5E4 6EE)	13.	(1 2)
7.	(1 6E E4)	14.	(6E EE)

For example, discrimination test number 1 requires that the program decide whether feature number 1 is present at a given location or whether there is an empty space. Recall that feature number 1 is "closed at the top". The discrimination between inner-feature number 1 and E was done with a convolution operator that was applied in a zone near the top of the two vertical parts in question. If a strong output was

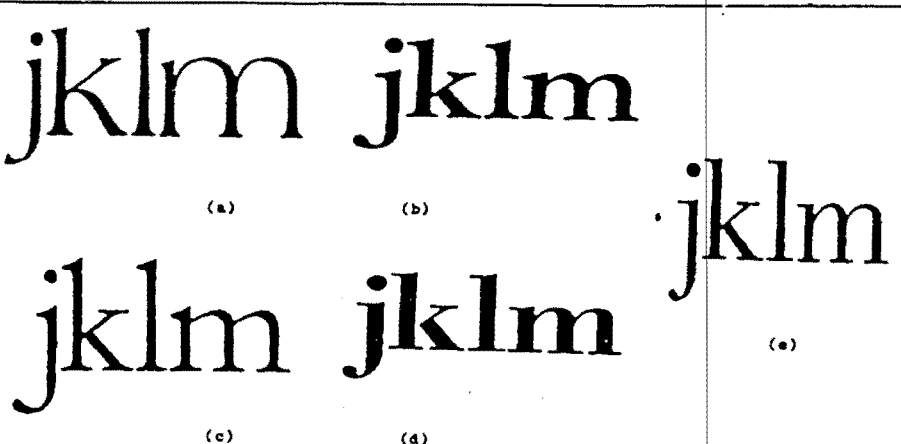


Figure 3. Examples of characters from each font used in the recognition tests. (a). Americana, (b). Baskerville Bold, (c). Bembo, (d). Bodoni Bold, and (e). Bodoni Book.

found, this was taken to indicate the presence of a closure. If no such output was found, E was decided. A similar procedure was derived for the other discrimination tests. Note that two consecutive 'E's enclose one of the letters that have a visual description of just 'O'.

The results of the recognition tests are shown in Table 3. Up to 97% of the input words were correctly recognized. However, this can range as low as 92% for the Bodoni Bold font. Since no reject option was programmed, the error rate is 100 minus the correct recognition rate. Work is currently underway on the refinement of the inner-features and the development of a streamlined procedure for carrying out the tests. This is expected to lead to much better performance.

Font	%correct	%error
Amer.	97	3
Bas.Bold	95	5
Bembo	97	3
Bod.Bold	92	8
Bod.Book	97	3

Table 3. Results of recognition experiment for the top 100 most frequent words in the Brown Corpus generated in five different fonts. The reading for pleasure strategy was used.

5. DISCUSSION AND CONCLUSIONS

A two-stage algorithm was presented for reading images of words that consists of hypothesis generation followed by hypothesis testing. The design of this algorithm was based on the human word recognition process. The hypothesis testing stage was the primary subject of this paper.

The hypothesis testing process was formulated as a tree search interpretation of an input image, where the search space was determined from the results of the hypothesis generation stage. The hypothesis testing process uses tests at each node that discriminate between a small number of features. If a test succeeds, the number of words that could exist in the input are reduced. If the test does not succeed, backtracking takes place and the search continues.

A statistical study of the trees used in hypothesis testing for vocabularies of from 10 to 3000 words was carried out. This study showed that the average number of words in any search space was always less than or equal to 4.0, although it did range up to 32 in one case. The number of trees with no solution was only about 2% of all possible trees. The shortest path in all the search spaces never exceeded three nodes. This says that no more than three tests need to be executed consecutively to recognize any input word.

The discrimination tests at every node were also studied. It was found, for the 3000 word vocabulary, that there were only up to 923 different tests in all parts of every tree and only 212 different tests in a collection of shortest paths from every neighborhood. Therefore, at most three tests would have to be executed to recognize any of 3000 different words and these tests would be chosen from among only 212 different possibilities.

Image processing experiments were also carried out that showed the viability of this methodology. Images of 100 different words in five different fonts were input to this technique. Correct recognition was achieved in 92% to 97% of all cases.

These results show that it is advisable to design a reading algorithm by application of knowledge about the human reading process. The algorithm presented in this paper concerned only two components of this process that deal with word level visual information. The complete human reading process contains many other stages such as syntactic and semantic analysis that should be a topic of further research.

Acknowledgements

The authors gratefully acknowledge the support of the United States Postal Service under the BOA program.

References

1. M. Brady, "Toward a computational theory of early visual processing in reading", *Visible Language XV*, 2 (Spring 1981), 183-215.
2. J. K. Cattell, "The time it takes to see and name objects", *Mind 11* (1886), 63-65.
3. L. D. Earnest, "Machine recognition of cursive writing", *Information Processing 1962, Proceedings of IFIP Congress 62*, Munich, Germany, August 27 - September 1, 1962, 462-466.
4. J. J. Hull, S. N. Srihari and R. Choudhari, "An integrated algorithm for text recognition: comparison with a cascaded algorithm", *IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-5*, 4 (July, 1983), 384-395.
5. J. J. Hull, "Word shape analysis in a knowledge-based system for reading text", *The Second IEEE Conference on Artificial Intelligence Applications*, Miami Beach, Florida, December 11-13, 1985, 114-119.
6. H. Kucera and W. N. Francis, *Computational analysis of present-day American English*, Brown University Press, Providence, Rhode Island, 1967.
7. D. Marr, *Vision*, W.H. Freeman and Company, San Francisco, 1982.
8. G. Nagy, "Optical character recognition: theory and practice", in *Handbook of Statistics*, vol. 2 , P. R. Krishnaiah and L. N. Kanal (editor), 1982, 621-649.
9. K. Rayner, "The perceptual span and peripheral cues in reading", *Cognitive Psychology 7* (1975), 68-81.
10. R. J. Shillman, *Character recognition based on phenomological attributes: theory and methods*, Massachusetts Institute of Technology, August, 1974. Ph.D. Dissertation.
11. R. Shinghal and G. T. Toussaint, "A bottom-up and top-down approach to using context in text recognition", *International Journal of Man-Machine Studies 11* (1979), 201-212.
12. C. Y. Suen and R. J. Shillman, "Low error rate optical character recognition of unconstrained handprinted letters based on a model of human perception", *IEEE Transactions on Systems, Man, and Cybernetics*, June, 1977, 491-495.
13. M. M. Taylor, "The bilateral cooperative model of reading: a human paradigm for artificial intelligence", in *Artificial and Human Intelligence*, A. Elithorn and R. Banerji (editor), North-Holland, 1984.
14. J. R. Ullmann, "Advances in character recognition", in *Applications of Pattern Recognition*, K. S. Fu (editor), CRC Press, Boca Raton, Florida, 1982, 197-236.