The application of optical character recognition techniques to bandwidth compression of facsimile data

by PATRICE J. CAPITANT and ROBERT H. WALLIS
Compression Labs
Cupertino, California

INTRODUCTION

The goal of facsimile bandwidth compression is the efficient transmission of documents achieved by the removal of redundancy in the encoding technique. For the case of printed or typewritten documents, the most powerful encoding technique is the Combined Symbol Matching (CSM) algorithm, which is based on the detection of recurrent patterns (such as alphanumeric characters) in the document being encoded ⟨4,5,6,10⟩. As the transmitter scans the document, it locates and extracts isolated patterns, transmits them to the receiver, and stores them in a library. Using the received patterns, the receiver also accumulates an exact copy of the transmitter's library. As each new pattern is isolated, it is compared with the library patterns which have been previously encountered. If the pattern is unfamiliar, it is added to the library. However if a "match" is detected, this indicates a recurrence of a pattern, and there is no need to retransmit it, since it is available in the receiver's copy of the library. Therefore, the library entry number (library ID) is transmitted instead, enabling the receiver to reconstruct the pattern from the "prototype" in its library. Since the library ID can be transmitted with far fewer bits than the binary pattern that it points to, a significant bandwidth compression may be attained. For printed documents, the CSM algorithm is typically twice as efficient as the best run-length coding algorithms.

In order to operate efficiently, the transmitter must not allow very many recurrences to escape detection, since this leads to a loss of compression. Conversely, it must also avoid declaring dissimilar characters to be a match, since this leads to a substitution error in the reconstructed document. This paper deals with the way in which the CSM algorithm determines whether two patterns match or not, and how the basic algorithm may be modified to perform optical character recognition.

COMBINED SYMBOL MATCHING SYSTEM

The Combined Symbol Matching (CSM) system is a dual mode encoding system that possesses the advantages of extended run-length encoding and symbol recognition systems. Figure 1 illustrates the block diagram of the encoding system. In operation, a number of scan lines (equal to about two to six times the average character height) of binary image data are stored in a scrolled buffer. This data is then examined line-by-line to determine if a black pixel exists. If the entire line contains no black pixel, the information is encoded by an end of line code. On the other hand, if a black pixel exists, a blocking process is conducted to block the symbol. For those blocked symbols, further processing is required to determine if a replica of the symbol in question already exists in the library. This process involves the extraction of a set of features, a screening process to reject unpromising candidates, and finally a series of template matches. The first blocked symbol and its feature vector are always put into the prototype library, and as each new blocked symbol is encountered, it is compared with each entry of the library that passes the screening test. If the comparison is successful, the library identification (ID) code along with the location coordinates are transmitted. If the comparison is unsuccessful, the new symbol is both transmitted and placed in the library. Those areas in which the blocker cannot properly block the symbol are assigned to a residue, and a two dimensional run-length coding technique is used to code the residue data.

The following sections summarize the compression and expansion algorithms.

Compression

The compression technique is based on the following sequence of operations.

1. The raster image is scanned, one line at a time, until a black pixel is found. This is called a "key pixel."
2. The local area around the key pixel, called a "trial block," is examined to isolate a small contiguous group of black pixels roughly the size and shape of an alphanumeric character.
3. If no symbol is found within the trial block, the trial block is left as a residue. The blocker is designed to avoid the residue in subsequent search for key pixels.
4. Residues are encoded using a two dimensional run-length code ⟨8,9⟩.

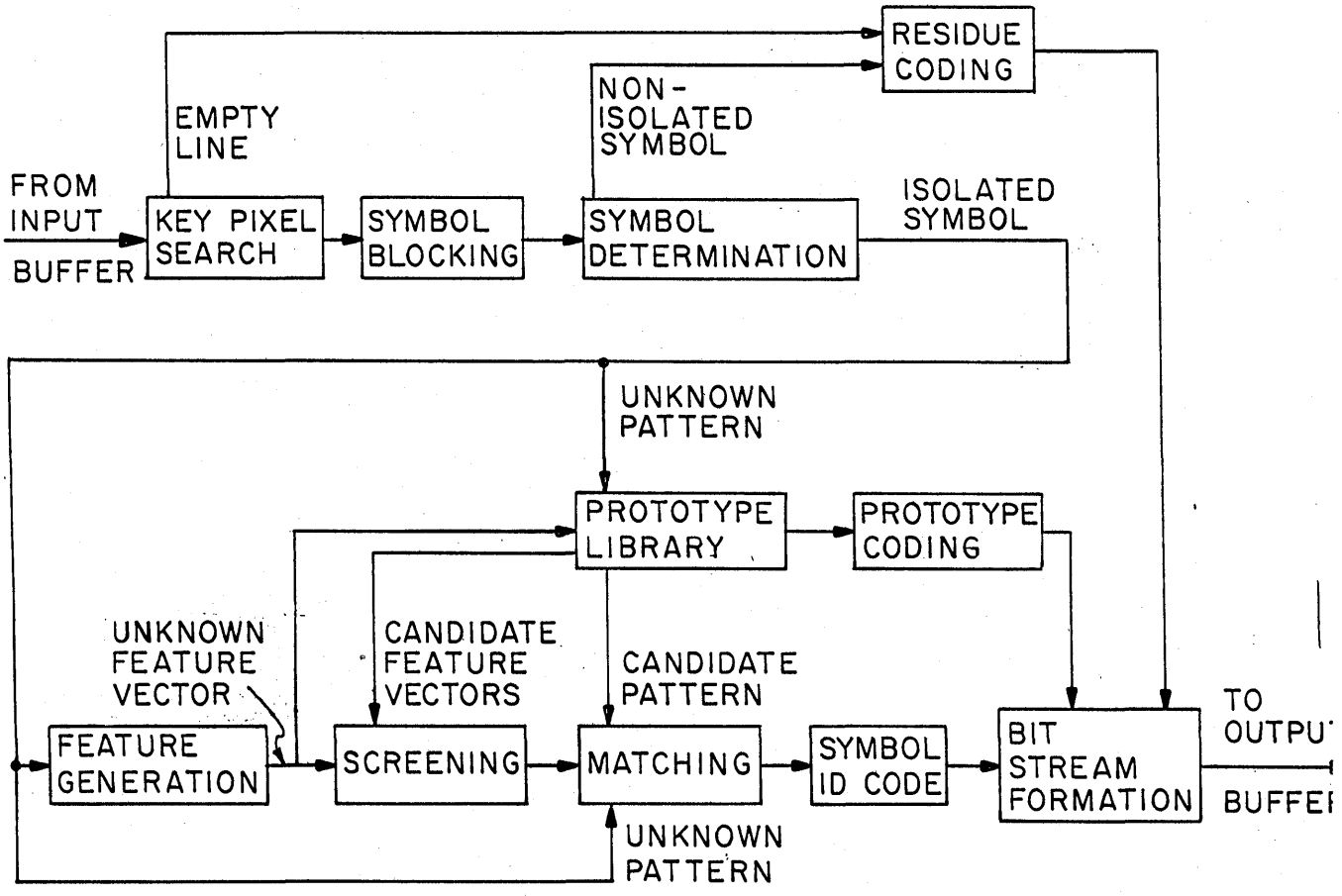


Figure 1—Combined symbol matching facsimile coder.

5. Symbol blocks are isolated, labeled, and removed from further key pixel consideration. The blocks are used to build a library against which other symbol blocks may be matched. The library is built from scratch with each new page, and updated as new symbols are found.
6. A set of features is measured for each symbol block. These are also stored in the library.
7. As each symbol is found, its features are compared to the features of symbols in the library. Then library entries with features most like the new blocks are treated as candidates for the matching process.
8. Library candidates are aligned with the trial symbol block, and a pixel-by-pixel comparison is made.
9. If the matching error is less than a threshold, the matching process is stopped, and the identification of the matching library member is stored for later coding.
10. If no match is found, the block is stored in its entirety as a "prototype" block. It is also entered into the library.
11. When a line has been completely processed, i.e., all pixel patterns in the line have been labeled as either residue or symbols, the code bits for the residue are concatenated with the code bits for the symbol blocks, a linesync code word is added every K lines, and the resulting sequence is transmitted as the data stream.

Expansion

The expansion technique is based on the following sequence of operations.

1. Portions of the input data stream carrying the code for one line of the image are isolated. The lines are separated by special "end of line" code words.
2. The code format is used to separate residue from the symbol code. These codes are processed separately.
4. The symbol code is used to generate a library of symbol blocks, and the ID code is used to select library entries.
5. The residue and symbol blocks are combined.

System elements

The following sections describe the key elements of the CSM system.

Input Buffer

The facsimile data to be processed is stored in a scrolled buffer that "scrolls" through the input document. This is accomplished by rotating the addressing of the input memory such that the newest line that enters the buffer is written over the oldest line. The buffer contains 128 lines, which is about four times the height of the largest character that can be matched.

Symbol blocking

The function of the key pixel scanner Symbol Blocker is to examine the input buffer in a systematic fashion, and to locate the position and size of any isolated symbols. A pixel in the buffer, denoted here by the character '@,' is considered to be a *key pixel* whenever it is black and the four neighbors located above it and to its left are white, as shown.

```
...
.@
```

Whenever a key pixel is encountered, the blocker is initiated. It will try to extract the symbol connected to the key pixel at its top and is delimited by a border of white pixels. If such a symbol can be found, it has to fit with its border into a 32 × 32 array. If no connection is made, go to the next key pixel. If a connection is made, the symbol is stored with its borders in a 32×32 RAM and erased from the input buffer.

The blocker algorithm separates the symbol from its surroundings by determining its boundary (perimeter). It does this by starting on a known exterior point (i.e. the key pixel) and following the exterior of the symbol in (say) the clockwise sense until the entire perimeter has been traced. Consider the eight nearest neighbors of a central pixel to be indexed as follows:

```
5 6 7
4 X 0
3 2 1
```

A clockwise rotation from a perimeter point (i) is given by

$$j=(i+n)(\text{mod } 8)$$

where n is the number of clockwise 45 degree increments in the rotation. The perimeter following algorithm consists of searching through the boundary points in a clockwise sense until a black pixel is encountered. The search is initiated at the neighbor that corresponds to the previous boundary point.

Assuming "X" in the above diagram is the key pixel of a possible symbol in a trial block, the algorithm first rotates counter-clockwise from index four until a black pixel is encountered. This vector is stored, and utilized as the stopping criterion, since the last vector in the clockwise trace is exactly 180 degrees out of phase with the first vector in a counter clockwise trace. In terms of the above notation

LAST(CLOCKWISE) = FIRST(COUNTER-CLOCKWISE) − 4 (mod 8)

Once the perimeter has been traced, the character must be extracted from the page for processing. This is accomplished by generating a "mask" which contains all the interior points of the perimeter, and performing a Boolean "and" between the mask and the trial block. The mask may also be used to erase the processed character from the document so that the residue contains only unblocked patterns.

The generation of the mask is based on the relation between a closed curve and its internal area. Specifically, the line integral gives:

$$A=\oint y\,dx$$

where A is the internal area.

A discrete counterpart of the continuous line integral expression has been developed which is amenable to digital mechanization. Each link of the perimeter is specified by a chain code in the range (0,7), and thus representable as a 3 bit code. The following algorithm, which starts with a *blank* buffer generates a mask of all the interior points of the perimeter:

(1) Complement all locations to the left of the key pixel and use the vector whose destination is the key pixel as initial source vector.
(2) All locations are then determined by a source vector and a destination vector. If $N(s)=A\times4+B\times2+C$ is the direction of the source vector (bits A,B,C specify the chain code) and $N(d)$ is the direction of the destination vector, let $N(d)-1$ (mod 8) $=D\times4+E\times2+F$. Let

$R=A$ .nor. D
$L=A$ .and. D
$S=$(.not. E .or. B) .and. (A .xor. D)

Apply the following rules:

$R=$.true.⇒Complement location and locations to the left
$L=$.true.⇒Complement locations to the left
$S=$.true.⇒Complement location

After the entire perimeter has been followed the mask buffer is complete.

In addition, the area enclosed by the boundary may be easily calculated as the perimeter is being traced. It is given by

$$\text{AREA}=\sum_{J=0}^{NLINKS-1}\text{COL}(J)\,[\text{ROW}(J+1)-\text{ROW}(J)]$$

where NLINKS equals the number of lines in the chain of coordinate pairs representing the perimeter, and all indices are taken modulo NLINKS.

The area enclosed by the perimeter has proved to be a useful feature for symbol recognition. Specifically, the ratio of the perimeter squared to the area is invariant to magnification, rotation, and translation.

Feature Extraction

The most straightforward method to determine whether a match exists between an unknown symbol and one of the symbols stored in the library is to perform a template match between the unknown and every library symbol. However, a two dimensional template match is costly in terms of processing time. A method of reducing the number of such matches is required. The approach that has been taken is to extract a set of scalar "features" from the various symbols in the library. These features are used to reduce or "screen" the number of candidates for a template match to a tiny fraction of all the possibilities in the library.

The features used in the screening process are the block height, block width, perimeter length, and internal area.

Candidate Screening

The purpose of the screening process is to reduce the burden on the template matcher by passing only "good prospects" to the matcher. This is accomplished by calculating the feature space distance between the unknown and each library entry, and selecting the library candidate with the smallest distance as the best prospect for a match. If this match is rejected, the next best candidate is considered, and so forth, up to a maximum of N. The distance "metric" used to determine how "close" an unknown is to a particular candidate is the "city block" distance defined by

$$D(U,C) = \sum_{I=1}^{N} |F_C(I) - F_U(I)|$$

where $F_C(I)$ is the Ith feature of the candidate, $F_U(I)$ is the Ith feature of the unknown, $|*|$ denotes the absolute value, $D(U,C)$ is the distance between the unknown and candidate, and N is the number of features.

Template Matcher

The template matcher forms a comparison between the binary patterns of a detected symbol and a library prototype symbol. Consider a two-dimensional binary pattern represented by $A(C,R)$ where $C = 1,2,..,N$ and $R = 1,2,..,N$. A conventional template matcher calculates the similarity between a pair of vector patterns $A(C,R)$ and $B(C,R)$ by summing the number of picture elements (pixels) for which $A(C,R)$ and $B(C,R)$ differ. This "Exclusive Or" error is defined as

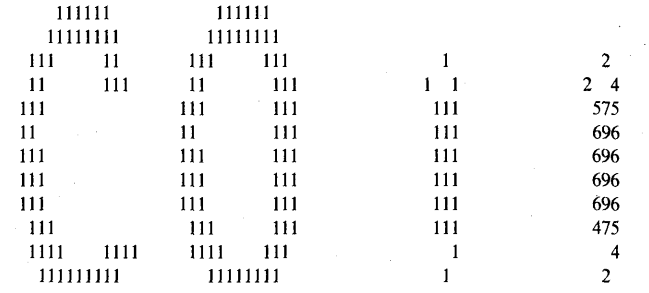
$$E = \sum_{C=1}^{N} \sum_{R=1}^{N} A(C,R) \oplus B(C,R)$$

where $\oplus$ denotes the Boolean Exclusive Or operation.

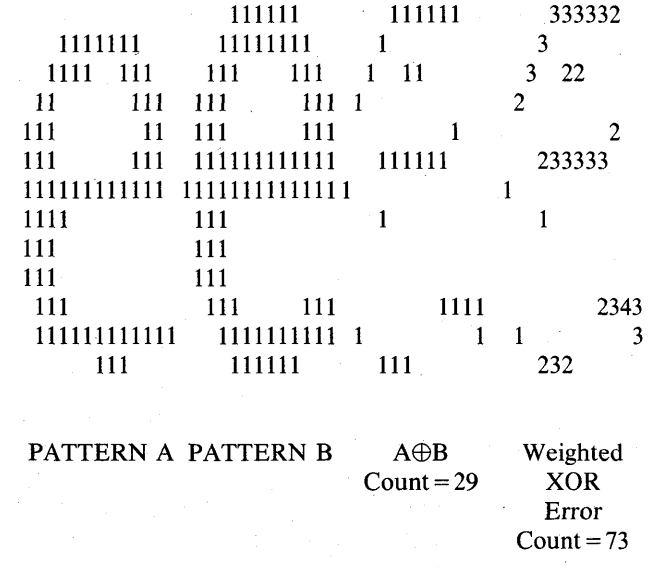
A major shortcoming of the conventional template matcher described above is that it treats all errors alike regardless of where they occur spatially. The improved matcher, to be described, utilizes an alternative error criterion that is based on the context in which the error occurred, known as the "Weighted Exclusive Or Count."

Weighted Exclusive Or Count

The motivation behind this error criterion may be appreciated by examining the Exclusive Or Error (denoted A⊕B) in the diagrams below:



Compare the Exclusive Or pattern for the "c" and "o" above with the pattern for the pair of "e's" below:



Note that the Exclusive Or Error count for the pair "c" and "o" (23) is actually *less* than that for the pair of "e's" (29) implying that by this error metric, "c" and "o" are "closer" than the pair of "e's" are to each other. However, the error pattern for the pair of "e's," which should be declared a match, is composed of *sparsely distributed* pixels, while the error pattern for the "o" and "c" shows a *dense node* of error pixels corresponding to the missing right segment of the "o." One way to quantify the density of such a "node" is to form a summation in which the "local den-

sity" of every black pixel is merely the sum of all the pixels in its 3×3 neighborhood if the pixel is 1, and 0 if the pixel is 0. The patterns above labelled "Weighted XOR Error" were calculated in this manner. Note that by this criterion, the associated counts indicate that the pair "c" and "o" are more separated (Count = 131) that are the pair of "e's" (Count = 73).

Optical Character Recognition

The CSM system can be modified to perform optical character recognition and various hybrid CSM/OCR tasks.

If a fixed set of symbols is to be expected, the library can be preconstructed. Unrecognized symbols would then be transferred in the residue.

A further modification would remove the residue coding subsystem and only transmit recognized symbols and their position on the page.

Finally the OCR mode would not transmit positions but only library codes, in the sequence which they would be read. In this mode, however, two extra subsystems need to be implemented to allow a meaningful reproduction.

Line Tracking

In the Western world, printed matter is "read" from left to right, and from top to bottom. Therefore, a symbol blocking system that transmits its output to a serial ASCII terminal must do the same. However, the CSM algorithm extracts characters from the document being scanned in a totally different fashion. As the line buffer scrolls through the page from top to bottom, the tallest of first encountered characters are removed from the document and processed through the recognition algorithm. Thus, characters emerge from the CSM process in a sequence which would be totally incomprehensible if viewed in chronological sequence. In the conventional CSM facsimile transmission mode, this is of no consequence, since characters are placed in their appropriate address locations regardless of their order of occurrence. In the serial symbol recognition mode, the transmitter will assign each character an ASCII code, assemble the codes into lines, inserting blanks, line-feeds, carriage returns, etc., and transmit the lines serially to the receiver. For single spaced or rotated documents, this "line-tracking" is more difficult than one would imagine. The problem is basically that of grouping the characters into lines. Determining the sequence in which they should be transmitted is relatively easy since the characters may be sorted by their column addresses. A significant benefit of this serial ASCII mode is that no information on character location need be transmitted since the correct sequence is all that is required in order to properly reconstruct the received document.

The line tracking algorithm is based on a straight line fit of the key pixel coordinates of characters on a text line, as illustrated in Figure 2. The straight line is defined parametrically as

$$R = S \times C + O$$

where R represents the row index, C is the column index, S denotes the text line slope, and O is its offset. As characters are encountered, they are assigned to the nearest straight line representing a text line.

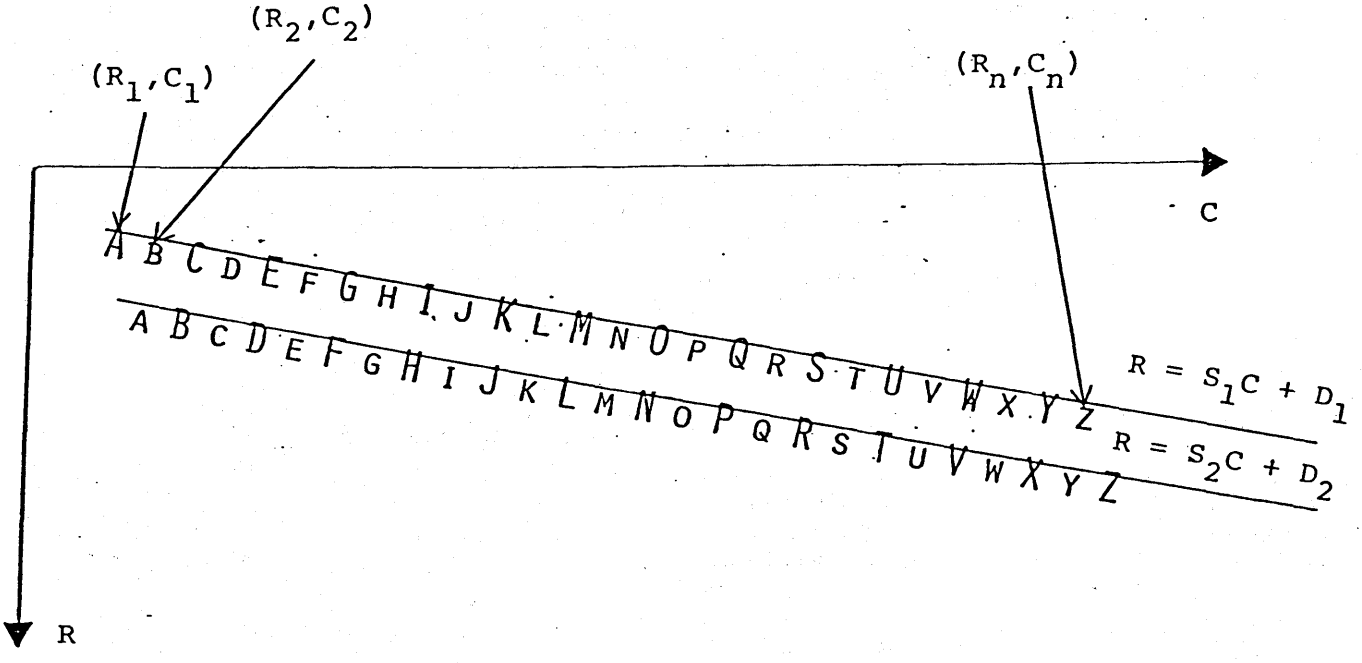


Figure 2—Line tracking.

COMPRESSION LABS, INC.

August 15, 1978

Telecommunications Manager
International Company
1111 Broadway
New York, N.Y. 10022

Dear Mr. Manager:

This letter will act as the standard for determination of the minimum compression ratios acceptable for the FAX-COMP, facsimile data compressor. The floppy disk of the FAX-COMP will be able to store at least nine copies of this page prior to overflowing which will guarantee a transmission time of less than 25 seconds for the page. This transmission time will be achievable using a 2400 baud digital modem for line connection.

Compression ratios of from 5:1 up to 25:1 can be expected from other pages of information, depending upon the actual content of the pages. These compression ratios are defined when using the 96 line per inch scanning resolution only.

Very truly yours,

Cloyd E. Marvin

CLOYD E. MARVIN
Vice President
Marketing

CM:vg

August 15, 1978

Telecommunications Manager
International Company
1111 Broadway
New York, N.Y. 10022

Dear Mr. Manager:

This letter will act as the standard for determination of the minimum compression ratios acceptable for the FAX-COMP, facsimile data compressor. The floppy disk of the FAX-COMP will be able to store at least nine copies of this page prior to overflowing which will guarantee a transmission time of less than 25 seconds for the page. This transmission time will be achievable using a 2400 baud digital modem for line connection.

Compression ratios of from 5:1 up to 25:1 can be expected from other pages of information, depending upon the actual content of the pages. These compression ratios are defined when using the 96 line per inch scanning resolution only.

Very truly yours,

?

CLOYD E. MARVIN
Vice President
Marketing

CM:vg

Figure 3—OCR results.

Handling of Special Characters

A number of characters which consist of two "sub-characters" must be treated as special cases in the symbol-matching mode. This is because the blocker/matcher would otherwise fragment them into their constituent parts and give misleading results. These characters are: (*i*), (*j*), (!), (:), (;), (=), and ("). After recognition of the two parts of the character, the system will check if two compatible symbols are on top or almost on top of each other. If so, the two symbols are merged into one. For example, two (.)'s on top of each other will be merged into a (:).

Performance

The CSM symbol recognition system has been extensively evaluated by computer simulation to optimize its performance and to determine its compression ratio with respect to other coding methods. The symbol recognition mode system has been tested with 86 sets of data, each containing 1,000 samples of one of the 86 symbols of the Courier 10 font. In these tests, no mismatches occurred, and only very badly damaged characters were rejected.

Figure 3 contains an example of a business letter and its reconstruction in the symbol recognition mode of operation. It should be noted that the reconstructed letter has been printed with a different font than the original; however, the format and spacing of the two letters are in basic agreement. The compression factor obtained for this document for operation of the CSM system in the symbol matching mode is about 257:1 and for operation in the facsimile mode is about 49:1.

REFERENCES

1. Arps, R. B., "Binary Image Compression," in *Image Transmission Techniques*, W. K. Pratt, Ed., Academic Press, New York, 1979.
2. Musmann, H. G. and Preuss, D., "Comparison of Redundancy Reducing Codes for Facsimile Transmission of Documents," *IEEE Transactions on Communications*, Vol. COM-25, No. 11, Nov. 1977, pp. 1425–1433.
3. Ascher, R. N. and Nagy, G. "A Means for Achieving a High Degree of Compaction on Scan-Digitized Text," *IEEE Transactions on Computers*, Vol. C-23, No. 11, November 1974, pp. 1174–1179.
4. Pratt, W. K., Chen, W., and Reader, C., "Block Character Coding," *Proceedings SPIE*, August 1976, pp. 222–228.
5. Chen, W., Douglas J. L., and Widergren, R. D., "Combined Symbol Matching—A New Approach to Facsimile Data Compression," *Proceedings SPIE*, August 1978, pp. 2–9.

6. Chen, W., Douglas, J. L., Pratt, W. K., and Wallis, R. H., "A Dual Mode Hybrid Compressor for Facsimile Images," *Proceedings SPIE,* August 1979.
7. White, H. E., Lippman M. D., and Powers, K. H. "Dictionary Look-Up Encoding of Graphics Data," in *Picture Bandwidth Compression,* T. S. Huang and O. J. Tretiak, Eds., Gordon and Breach, New York, 1972, pp. 265–281.
8. Mitchell, J. L. and Goertzel, G., "Two-Dimensional Facsimile Coding Scheme," *Proceedings International Communications Conference,* 1979, pp. 8.7.1-8.7.5.
9. "Proposal for Draft Recommendation of Two-Dimensional Coding Scheme," *Report of Study Group XIV,* Contribution No. 42.
10. Pratt, W., Capitant, P., Chen, W., Hamilton, E., and Wallis, R., "Combined Symbol Matching Facsimile Data Compression System," *Proceedings IEEE,* May 1980.