Document Image Analysis Using a New Compression Algorithm

Shulan Deng[1], Shahram Latifi, and Junichi Kanai[2]

[1] Department of Electrical and Computer Engineering
University of Nevada Las Vegas, Las Vegas, NV 89154-4026
{deng,latifi}@ee.unlv.edu
[2] Panasonic Information and Networking Technologies Laboratory
PINTL, 2 Research Way, Princeton, NJ 08540, USA
kanaij@research.panasonic.com

Abstract. By proper exploitation of the structural characteristics existing in a compressed document, it is possible to speed up certain image processing operations. Alternatively, one can derive a compression scheme which would lend itself to an efficient manipulation of documents without compromising the compression factor. Here, a run-based compression technique is discussed for binary documents. The technique, in addition to achieving bit rates comparable to other compression schemes, preserves document features which are useful for analysis and manipulation of data. Algorithms are proposed to perform vertical run extraction, and similar operations in the compressed domain. These algorithms are implemented in software. Experimental results indicate that fast analysis of electronic data is possible if data is coded according to the proposed scheme.

1 Introduction

In a large scale document conversion operation, paper documents are scanned, the quality of the scanning process is assured, and the document images are analyzed. Document images are usually compressed before being archived and analyzed. Traditional document analysis systems decompress page images completely before analyzing them. Systems that extract useful information in the compressed domain result in significant computational savings.

In the CCITT (newly renamed as ITU) Group 4 domain, Spitz [10] adapted the Baird's alignment technique to look for the pass modes, which are heavily populated at the fiducial point position of data, for detecting skew. Maa [6] observed that bars and space of barcodes are upright and should be perfectly aligned. Hence barcodes can be identified by finding the vertical mode codewords in the compressed domain. Recently Hull [5] compared the locations of pass modes contained in two bounding boxes for possible match while addressing the problem of document image matching.

In this paper, we will modify coding rules of the Group 4 and then derive a new coding scheme called Modified G4 or MG4, which gives us the flexibility for

S.-W. Lee and Y. Nakano (Eds.): DAS'98, LNCS 1655, pp. 32–41, 1999.

storage and processing. In our vision, such an algorithm is useful as an internal compression method of a document processing system. A filter is used to convert the compressed data into one of the standard formats, such as CCITT G4, to maintain compatibility with other systems. This paper is organized as follows: an introduction of MG4 is presented in Section 2. Section 3 analyzes the MG4's performance and presents the experimental results. Based on the MG4 scheme, some algorithms for extraction of vertical runs, connected component extraction, and skew detection are proposed and results are given in Section 4. Section 5 concludes the paper.

2 Modified Group 4 (MG4)

The Group 4 [3] encoding is a two-dimensional coding, in which each coding line is coded with respect to the previous line, "reference line". Since scan lines in a document file are highly correlated, we can take advantage of this characteristic to explore pixel level structures within several adjacent lines. The coding scheme for the Group 4 includes three coding modes: pass mode, vertical mode, and horizontal mode. These modes are defined by relationship among five changing picture elements. In Group 4, an object of interest cannot be easily discriminated from the background in compressed domain. It is also difficult to perform other image analysis in this domain. To overcome this deficiency, a modified Group 4 coding scheme (MG4) is proposed.

We regard the image as black objects on a white background, which is the case for most applications. This new coding scheme is designed based on the following three considerations:

- To encode the data based on runs for fast processing
- To keep the original structure in the compressed domain as much as possible
- To reduce two-dimensional redundancy to improve compression

The scheme is based on the notions of reference run and coding run. Let b_1, b_2 denote the coordinates of the endpoints of the run in the reference line and a_1, a_2 denote the coordinates of the endpoints of the run on the coding line. If the intervals $[b_1, b_2)$ and $[a_1, a_2)$ have overlapping projections, i.e., the following inequalities hold:

$$a_1 \leq b2 \quad and \quad b1 \leq a2 \tag{1}$$

then we call $b_1 b_2$ and $a_1 a_2$ the reference run and the coding run, respectively. A coding run is the run to be currently encoded. If a coding run has more than one reference run, then for simplicity, the leftmost reference run is chosen to encode the coding run. We define three modes for MG4 coding scheme: new vertical mode, new pass mode, and new horizontal mode.

2.1 Vertical Mode in MG4

The definition of vertical mode in Group 4 is extended here. In the MG4, if a coding run has a corresponding reference run, then the coding run can be represented by vertical mode. In other words, this mode occurs when the run on

reference line has overlapping pixels with the coding run, i.e., satisfy Ineq. 1. In this mode we have two parameters with sign to encode; these are represented as $V(a_1b_1, b_2a_2)$ (see Fig. 1).

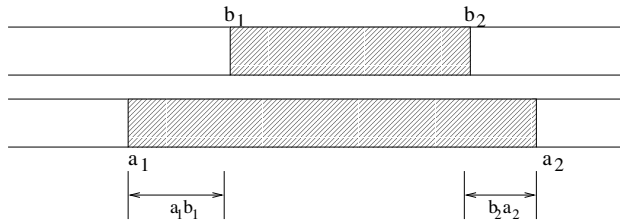


Fig. 1. MG4: Vertical mode

2.2 Pass Mode in MG4

New pass mode can be denoted as $P(n)$, where n is a non-negative integer. When n is 0, the coding run uses the same reference run as the last coding run (see Fig. 2). When n is nonzero, the case in Fig. 3 occurs. In this case, n records the number of runs skipped on reference line, then a vertical mode can be employed to encode the coding run.

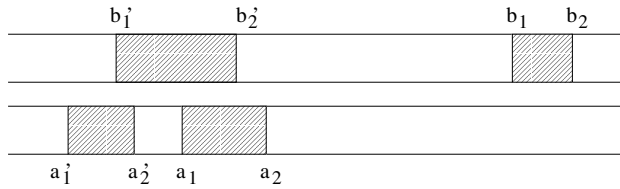


Fig. 2. MG4: New pass mode P(0)

Suppose the two runs $a'_1a'_2, a_1a_2$ in Fig. 2 are to be encoded. Since the run $b'_1b'_2$ satisfies the Inequality 1, it is the reference run of the coding run $a'_1a'_2$. So the $V(a'_1b'_1, -a'_2b'_2)$ is used to encode $a'_1a'_2$. Then next coding run is a_1a_2. It is seen that $b'_1b'_2$ still satisfies the Inequality 1 and is the reference run of a_1a_2. The modes $P(0), V(-b'_1a_1, b'_2a_2)$ are the representation of a_1a_2 in the compressed domain. For the same reason, we encode the case of Fig. 3 as $V(a'_1b'_1, -a'_2b'_2), P(2), V(a_1b_1, -a_2b_2)$. Because of the pass mode, two dimensional correlation can be utilized to maximum extent while compressing an image. So the pass mode implies compression efficiency and structure information.

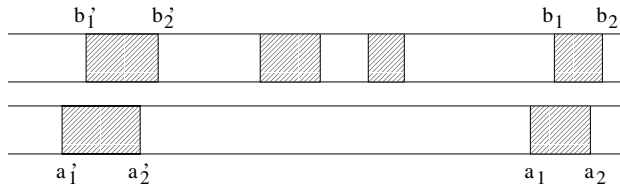


Fig. 3. MG4: New pass mode P(2)

2.3 Horizontal Mode in MG4

This mode occurs when a reference run cannot be found for the coding run. It represents a situation that the run just above the coding run on reference line is a white run. Obviously it implies the beginning of local characteristic region or connected component. The white run and black run are coded as $H(a'_2a_1, a_1a_2)$ (see Fig. 4).

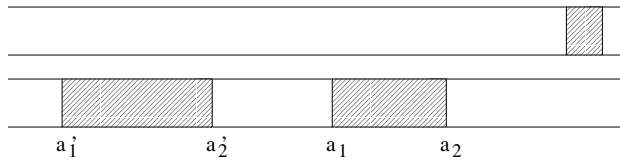


Fig. 4. MG4: New horizontal mode

The MG4 code is based on the modified Huffman scheme. Every mode's code has two parts: a flag indicating the mode, and a value or two values representing the length of the black run or white run. These values are taken from the white-run and black-run code tables of one dimensional coding.

3 Performance Analysis

MG4 is a statistical type coding in which highly probable symbols will get short codes. In this section, we will derive an estimation for the compression ratio that the MG4 yields, and present experimental results.

3.1 Compression Ratio Estimation

Compression ratio for MG4 is given by:

$$CF \approx \frac{Total\ \ Number\ \ of\ \ Pixels\ \ in\ \ Image}{n_v\bar{L}_v + n_p\bar{L}_p + n_h\bar{L}_h} \tag{2}$$

where n_v, n_p, n_h denote the number of new vertical mode, new pass mode, new horizontal mode, respectively, and $\bar{L}_v, \bar{L}_p, \bar{L}_h$ are the average lengths of corresponding modes in an image. In deriving Eq. (2), we have ignored the size of end-of-line, and end-of-page codewords

In MG4 coding scheme, as there are three coding modes, the entropy of MG4 can be represented as:

$$H_{MG4} = \sum_{i=1}^{3} P_i H_i \tag{3}$$

where P_1, P_2, P_3, denote these three modes' probabilities, and H_1, H_2, H_3 correspond to their entropies.

Starting with different ideas, we apply the approach used by Huang [4] to analyze the performance of MG4. It is noted that our model used is different from Huang's Predictive Differential Quantizing (PDQ). In the PDQ, the quantities Δ' (changes in white-to-black transition locations) and Δ'' (changes in the black runlengths) are transmitted, together with the New start and Merge messages indicating the start and the end of a black area. On the other hand, in our model, the three modes of vertical mode, pass mode, and horizontal mode are used. Because of the similarity of MG4 and PDQ, the derivations will be identical under our assumptions, and result are cited here.

First let us look at the vertical mode, which has two parameters a and b, where a is the differences in white to black transition and b is the difference in black to white transition compared with reference run. Suppose H_a is the entropy of a, then it is found that H_a is equal to 3.5 [4]. Depending on the type of image, the distribution of the other parameter, b, can be completely dependent on the a or completely independent of a. And the the entropy of b, H_b, satisfies: $0 \leq H_b \leq 2H_a$. To simplify the analysis, assuming that H_a=H_b. Also, suppose the number of pass mode and horizontal mode is much smaller than the number of vertical mode, and the number of vertical mode is approximately equal to the number of runs. Then according to Eq. (2) and the above assumptions, compression ratio can be roughly estimated as: $CF = (Average\ \ Run\ \ length)/3.5$.

3.2 Experimental Results

This new coding scheme is based on the international A4 paper-size standard. In this experiment, we chose CCITT1-CCITT8, the *de facto* standard test images, as test files each digitized at 200 dpi and converted into binary image with the size of 513229 bytes. The new coding scheme is implemented in software on a SGI O2 workstation under Unix environment. Experimental results shows the actual average occurrence probability of the three modes: 86% (vertical), 9% (pass) and 5% (horizontal). We can see that the vertical mode occurs with a much higher probability than the other modes. It is found that MG4 achieves high compression which is close to Group 4. On the average, compression ratio is about 17:1.

From the experiment, we found that the average run-length is 45.532 pixels. According to above CF formula, the compression factor is equal to 13. We

assume H_a=H_b while deriving the formula. In fact, in the test image set, the H_b is less than H_a because of the correlation between a and b, so it should not be surprising that the estimated compression factor is smaller than the experimental results.

4 Document Analysis

An image analysis system realizes the objective of understanding document by using different algorithms or techniques in different stages of analysis. In the following subsections, we will discuss several subsystems of document analysis which can operate directly in MG4 domain: vertical run extracting systems, connected component extraction, and skew detection.

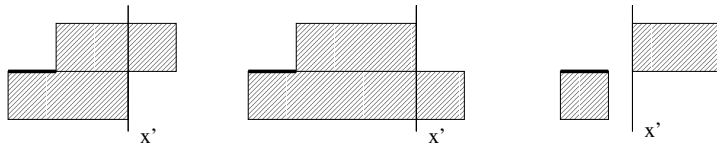
4.1 Extraction of Vertical Runs

Extraction of vertical runs from compressed image data is useful for operations such as rotation, and smearing [11]. Shima et al. [9] present a method of converting horizontal runs into vertical runs while rotation of the binary image based on coordinates of data for the start and the end of the run is executed. Yet, in this method, every black pixel is accessed in order to extract the vertical run, which increases computational complexity.

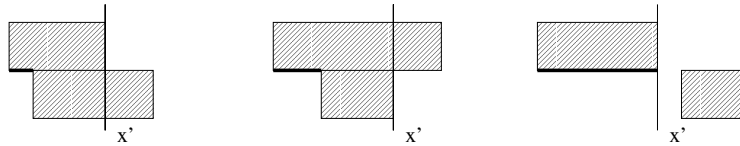
We notice that the beginning and ending points of each vertical run can be also detected while a MG4 compressed data is symbol decoded. The characteristics of MG4 are utilized to speed up the procedure of vertical run extraction by examining the modes encountered . As mentioned before, in vertical mode, if a_1 is to the left of b1, it will be the beginning of vertical runs; if a_1 is to the right of b1, it will be the ending of vertical runs. $V(0,0)$ can be ignored. For the pass mode ($n > 0$), the reference runs skipped are the end of vertical runs. And a horizontal mode always indicates the beginning of vertical runs. If the reference line above the coding run is black, the gap between the coding runs indicates the ending of vertical runs.

In Fig. 5, six typical relative positions of two runs on adjacent lines are shown. Figure 5.A presents three cases of detecting the beginning of vertical run, in which the starting position of the run on coding line is located to the left of that of the run on the reference line. The black thick line stands for the areas of vertical run's beginning. In every case of Fig. 5.B, the starting position of the run on coding line is located to the right of that of the run on the reference line and the ending points of the vertical run are marked by black thick lines. X' in Fig. 5 is the relocated starting point for next comparison.

Obviously, for the beginning of every vertical run, there always exist corresponding end points. We can easily get them and more importantly, get this information efficiently. For the document image with high resolution, whose average length of black runs is relatively large, the improvement in speed will be significant because the algorithm operates directly on the rundata.



A. Black thick line indicating the beginning points of vertical run



B. Black thick line indicating the ending points of vertical run

Fig. 5. Finding the beginning point and ending point of vertical run

4.2 Connected Component Extraction Analysis

An important part of the image analysis is to decompose the image into entities (object, region, and blob). Connected components are extracted and used in a number of ways, such as calculating the number of objects, object matching and analysis of components. In traditional methods, the extraction of components is often done by using 2×2 [2] or 3×3 window to scan the image for contour detecting, making it time-consuming. MG4 is actually a differential coding scheme which keeps the information about the object's contour. Because of this characteristic, we can organize useful information in compression domain to do the component extraction easily.

We must mention that Pavlidis [8] extends the result of a hybrid vectorization algorithm and describes a thinning method (LAG) to operate directly on the run length encoding of bilevel image. There are some similarities between our algorithm and Pavlidis' because both methods are proposed to process run data. However, our analysis and algorithm are based on MG4, a new coding scheme designed for easy processing in compressed domain. Therefore, our algorithms are essentially different. In MG4, the horizontal mode often indicates the starting point of a component. Sometimes there are several horizontal modes for a single connected component which eventually are connected by a pass mode with nonzero parameter. When the vertical mode is encountered, the current run is part of a component. If a pass mode with parameter 0 occurs, next run coded by vertical mode is also connected with the component.

Extraction of connected components can be performed using steps below (see Fig. 6):

1. The information of two adjacent scan lines is always renewed and kept after finishing the processing of every scan line.

2. The horizontal code is searched in the compressed domain. If a horizontal code whose black run represents the beginning of a component is found, then a database is generated for the new component.
3. According to the particular property of each mode, parts of component on subsequent lines are extracted and saved into corresponding databases.
4. The ending of a component is indicated by the horizontal mode's white run or a pass mode with non-zero parameter on the next line.
5. The databases which belong to the same component are merged to form a single database by checking whether current coding run has other reference runs. Then this database contains information of a connected component.

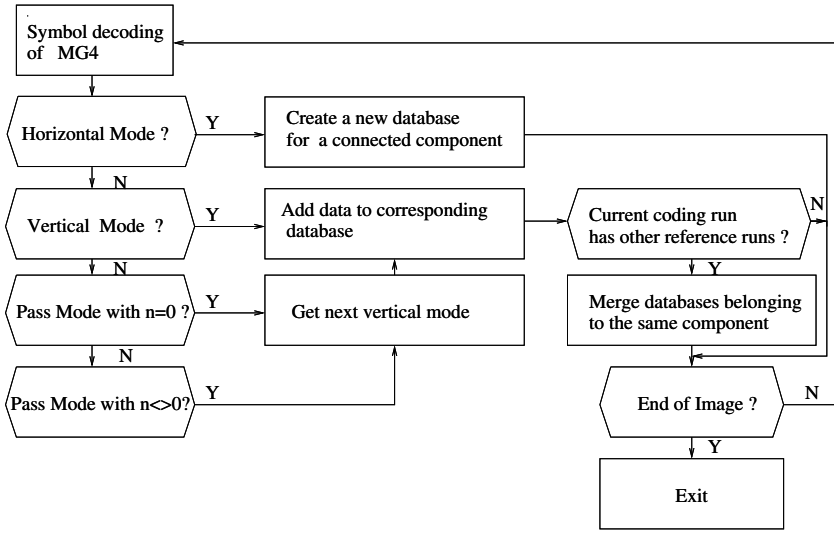


Fig. 6. MG4: Flow Chart of Connected Component Extraction

In our experiments, the complexity of algorithms is approximately of the order of the number of run length code segments. Also, components are extracted without decompressing the image. As a result, the processing speed is much faster than that in special domain.

4.3 Skew Detection

Skew detection is not a new topic. Many algorithms have been studied. Most of them was done in the spatial domain [1,7]. As we mentioned before, Spitz found the skew angle by detecting the occurrence of pass mode in the Group 4 domain. It turns out that in MG4 domain we have the same convenience to detect the skew angle as in the Group 4 domain. The principle is that the horizontal

mode often distributed at the top point of every character. Figure 7 illustrates the distribution of horizontal mode in a document image. Thus, many of these points are on or near the x-height line of a text-line. We measure the slope of extracted text lines according to the distribution of horizontal run to determine the skew angle. Combining the alignment-measure approach with locating the maximum techniques, we detect the skew angle in MG4 domain. Experiments show that the accuracy and the speed for estimating skew angle is comparable with that in Group 4 domain.

Fig. 7. Detecting the key points represented by horizontal mode for skew angle

5 Conclusion

In this paper, the processing of algorithms in compressed domain was reviewed briefly. Based on a coding scheme, MG4, we investigated the feasibility and efficiency of preserving the necessary structure in compressed domain to manipulate the encoded data directly. This coding scheme offers saving in storage and transmission time. The experimental results showed that our promises of efficient data compression of document image and easy image processing of coded data were realized.

Acknowledgment

This work was supported by NSF Research Grant IRI-9616206 and by NASA Research Grant NAG5-3994. Kanai worked on this project when he was research scientist at the Information Science Research Institute, University of Nevada, Las Vegas.

References

1. H. S. Baird: Proc. of SPSE Symp. on Hybrid Imaging Sys. Rochester, N. Y. **78** (1987) 21–24 39
2. N.Bartneck: Computing. **42** (1989) 17–34 38
3. CCITT Recommendation T.6, Facsimile Coding Schemes and Control Functions for Group IV Facsimile Apparatus, In Terminal Equipment and Protocols for the Telematic Services, Vol. VII, Fascicle VII.3, Geneva 1989 33
4. T. Huang: IEEE Transactions on Communication. 36

5. J. J. Hull and J. F. Cullen: Proc. Of 4th Intern. Conf. on Document Analy. and Recogn., Ulm, Germany. (1997) 308–312 32
6. C. Maa: Graphical Models and Image Processing. **56** 1994 352–356 32
7. Y. Nakano, Y. Shima, H. Fujisawa et al., Proc. of Intern. Conf. on Patt. Recogn. 1990 687–689 39
8. T. Pavlidis: Graphical Models and Image Processing. **35** 1986 111-127 38
9. Y. Shima, S. Kashioka, and J. Higashino: Systems and Computers in Japan. **20** 1989 91–102 37
10. A. L. Spitz: Proc. of the 1st Symp. on Document Analy. and Inform. Retri. 1992 11–25 32
11. F. M. Wahl, K. Y. Wong, and R. G. Casey: Computer Graphics and Image Processing. **20** 1982 375-390 37