Australian Journal of Basic and Applied Sciences, 9(11) May 2015, Pages: 893-903

ISSN:1991-8178

Australian Journal of Basic and Applied Sciences

Journal home page: www.ajbasweb.com

Complex Compound Document Compression using a Multidimensional Multiscale Parser Algorithm

[1]K.Uma, [2]P. Geetha, [3]A. Kannan, [4]Ramesh Kumar

[1,2] Dept. of Computer Science and Engineering, College of Engineering, Anna University, Chennai India,
[3]Dept. of Information Science and Technology, College of Engineering, Anna University, Chennai, India
[4]Dept. of Computer Science and Engineering, Sree Sastha College of Engineering, Chennai, India

ARTICLE INFO

Article history:
Received 16 March 2015
Accepted 28 May 2015
Available online 22 June 2015

Keywords:
scanned compound compression, Document compression, MMP, Multidimensional Multiscale Parser, Recurrent pattern matching.

ABSTRACT

Nowadays scanned document is a part of our daily life. It may be in different forms like e-journals, magazines, book copies. The generation of a large number of scanned documents arises the problem of efficiently coding them. It is difficult to perform operations such as upload, send, and download in a low bandwidth communication medium. That's why a lot of research is taking place in this stream looking for efficient methods to compress scanned documents. The compression is based upon a recently announced coding paradigm called multidimensional multiscale parser (MMP). MMP uses approximate pattern matching, with adaptive multiscale dictionaries that contain concatenations of scale versions of previously encoded image blocks. These features of MMP results in high coding efficiency for an extensive variety of image types, by effectively adjusting the input image characteristics. This flexibility makes MMP a good contender for compound image encoding. The suggested algorithm first classifies the image blocks as smooth (texture) and nonsmooth (text and graphics). Smooth and nonsmooth blocks are then compressed using different MMP-based encoders, altered for encoding both kinds of blocks. The adaptive use of these two types of techniques resulted in performance gains over the original MMP algorithm, further increasing the performance advantage over the existing image encoders for scanned compound documents, without compromising the performance for other image forms. By using two different approaches for smooth and non-smooth blocks, it encoded scanned documents more efficiently than existing methods such as MMP-II. This is achieved without dropping quality of the document.

© 2015 AENSI Publisher All rights reserved.

To Cite This Article: K.Uma, P. Geetha, A. Kannan, Ramesh Kumar, Complex Compound Document Compression using a Multidimensional Multiscale Parser Algorithm. ***Aust. J. Basic & Appl. Sci.,*** *9(11): 893-903, 2015*

INTRODUCTION

The generation of a large number of scanned document images arises the problem of efficiently coding them. When retrieved from the Internet, digital images take a considerable amount of time to download and use a large amount of computer memory. The basic idea behind this method of compression is to treat a digital image, as an array of numbers i.e., a matrix. Each image consists of a fairly large number of little squares called pixels (picture elements). The matrix corresponding to a digital image, assigns a whole number to each pixel. For example, in the case of a 256x256 pixel grayscale image, the image is stored as a 256x256 matrix, with each element of the matrix being a whole number ranging from 0 (for black) to 255 (for white). The JPEG compression technique divides an image into 8x8 blocks and assigns a matrix to each block. One can use some linear algebra techniques to maximize compression of the image and maintain a suitable level of detail.

Image compression applications make it easier to compress images. The compression tools are user friendly and can be used by anyone with minimal knowledge. The images are compressed just by selecting the images and setting the options. One even gets to choose the algorithms for compressing images and hence have the control over the output image. Image compression applications compress images quickly. Thus, they result in efficient utilization of time, memory and bandwidth.

Related work:

MMP Algorithm:

MMP is a general purpose compression algorithm that can be used to compress a large variety of data sources. MMP has some important features that distinguish it from other pattern matching algorithms. It uses multiscale approximate

Corresponding Author: K. Uma, Dept. of Computer Science and Engineering, College of Engineering, Anna University, Chennai India
E-mail: umakbe@gmail.com

block matching combined with an adaptive dictionary. Traditional pattern matching methods approximate each fixed size image block X^l using one code-vector Sl with the same dimensions as X^l. MMP uses scale adaptive pattern matching that allows matching between vectors of different dimensions. If an original vector X^l, of scale l (with dimensions $2^{\lfloor l+1/2 \rfloor} * 2^{l/2}$ dictionary of scale (with dimensions pixels), is to be approximated using one vector S^k of the dictionary of scale k (with dimensions $2^{\lfloor k+1/2 \rfloor} * 2^{k/2}$) the algorithm will use a 2-D scale transformation Tlk, in order to convert Sk into a scaled version Sl before performing the match.

The selection of the best dictionary block X^l to represent is based upon an R-D optimization function J (T), given by

J (T) =D (T) +χR (T)

Where is a Lagrangian multiplier that weights the relative importance of the rate R (T) required for the representation and the block distortion D (T).

After selecting the code-vector that minimizes the Lagrangian cost function, the algorithm segments the original block into two new blocks, $X_1^{l-1}$and X_2^{l-2}, each with half the pixels of the original block. The matching procedure is then recursively used on each subblock. The sum of the representation cost of the two halves is then compared with the cost of representing X^l with a single code-vector, in order to decide whether to segment or not the original block.

The optimal block partitioning is represented by a binary segmentation tree, which holds all the information needed to generate the final bit stream. The tree leaves correspond to no segmented blocks, X^l that are approximated by Si^l one individual code-vector, identified by its index .The scale is related to the blocks dimensions corresponds to the tree level the leaf belongs to. For example, n initial block size of 16*16 corresponds to a segmentation tree with a root at level l=8 and potentially spanning down to l=0, the lower level of the tree that corresponds to 1*1blocks.

To generate the final bit stream, the segmentation tree is encoded with a binary flag:

• "0": used to represent block segmentation;

• "1": indicating that the current subblock should not be segmented. This flag is always followed by the index used to encode the corresponding 1*1 subblock. This flag is not used for level 0, where no further division is possible.

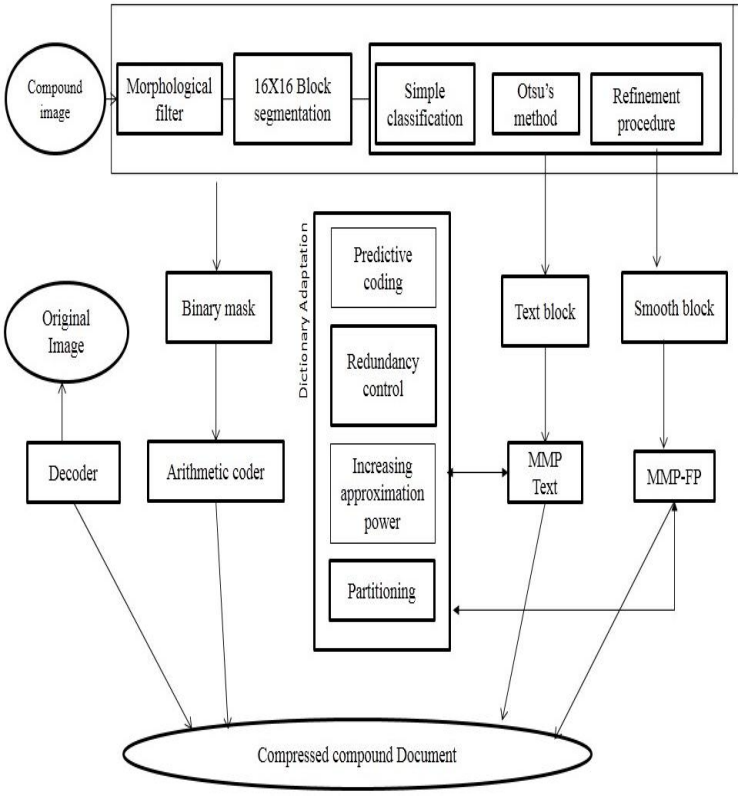

Fig. 3.1: System Architecture.

Proposed approach:

Proposed architecture includes four modules which are shown in figure 3.1. In which segmentation module divides the image into 16*16 blocks. Then we are applying morphological filters. A classification algorithm is used to classify them into text and smooth blocks. The second module can be dividing into two, MMP-FP and MMP-Text. Flexible partitioning is used to encode smooth image blocks while the MMP-text is used to encode text

blocks. An arithmetic coder is also used for better compression ratio. The decoder recreates original image at the receivers end.

Segmentation is used so that image is split into smooth and non-smooth blocks. Classification algorithm is used to obtain this. Two different versions of MMP algorithm MMP-FP and MMP-Text is then applied on these kinds of blocks separately to yield better results than original MMP. This algorithm used is dictionary based. Separate dictionaries will be kept on both senders and receivers side. This dictionary is used to reconstruct the image with a few blocks received from sender. This dictionary will keep on updated each time when it receives a new block of data from the sender. Arithmetic coder is used to encode the binary index generated as a part of compression process. The reverse should be applied in decoder inorder to generate the original one.

a. Image segmentation:

Input is any Scanned compound document. Image Segmented into 16*16 blocks of image. Apply morphological gray scale top-hat and bottom-hat filter operators. Top-hat operator allows identifying bright objects over dark background. Bottom-hat operator allows identifying dark foreground objects over a bright background. A block based classification algorithm is then applied to the enhanced images. The horizontal and vertical gradients of each 16 * 16 block are computed. Two thresholds are applied to the absolute value of these gradients in order to classify its pixels as low-medium- or high-valued. The gradient pixels of text blocks tend to be medium to high-valued. The gradient pixels of smooth blocks tend to be medium to high-valued. Expected output image which is segmented and each segment are classified into text and smooth blocks.

b. MMP-FP:

Input is smooth image blocks. Selection of the best dictionary block is based on the optimization function J (T) =D (T) +χR (T) where χ is the Lagrangian multiplier. Select the code vector which is having minimum cost. A tree is constructed based on cost. Predictive coding is done. Using coded samples to generate predictive blocks. It is subtracted from the original to get residual block. After checking with different predictive modes the algorithm chooses the best one. Then applying dictionary adaptation improvement techniques. Expected output is Encoded smooth image blocks.

MMP uses scale adaptive pattern matching that allows matching's between vectors of different dimensions. This technique also includes predictive coding and dictionary adaptation techniques. Predictive coding techniques have the well-known property of generating residue samples with highly peaked probability distributions centered on the value zero that favors the arithmetic coder adaptation process to the source. The improved dictionary adaptation results from the use of prediction error data.

c. MMP-Text:

The Input is Text blocks of original image. First selecting the best dictionary block is based on the optimization function J (T) =D (T) +χR (T) where χ is the Lagrangian multiplier. Select the code vector which is having minimum cost. A tree is constructed based on cost. Then applying other dictionary adaptation improvement technique. Output is coded text document.

d. Arithmetic Coder:

The input is binary segmentation mask. Each event in the file is having two steps. Then current interval is subdivided into subintervals, one for each possible event. The size of an event's subinterval is proportional to the estimated probability that the event will be the next event in the file, according to the model of the input. Select the subinterval corresponding to the event that actually occurs next, and make it the new current interval. Output enough bits to distinguish the final current interval from all other possible final intervals. Expected output is encoded binary.

e. Improving dictionary adaptation:

1. Improved context modelling for the code-vectors' indices, resulting in improved performance of arithmetic coder. The dictionary elements are organized into partitions, and each code-vector is identified using its partition (context) followed by its index inside that partition. The original block scale is used as a context, exploiting the fact that blocks generated at different scales have different matching probabilities.
2. An efficient redundancy control scheme for dictionary elements. The insertion of a new block in the dictionary is only performed if its distance relatively to an existing code-vector is inferior to a given threshold. This avoids the creation of new dictionary indexes corresponding to blocks that bring little distortion gains, with costs in increasing the average entropy of the indexes' symbols.
3. In order to improve the dictionary approximation power, MMP-II uses extra blocks, originated by geometric transformations and translations of the original block, to update the dictionary.
4. A norm-equalization procedure that allows the algorithm to adapt the new code-vector patterns to the statistical distribution of the residue signals.

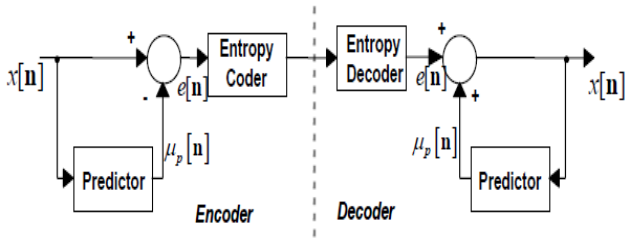

Fig. 3.2: Predictive Coding.

f. Predictive coding:

1. Prediction μ_p [n] is calculated for x[n] from previous samples $X_{N+n.}$
2. MMP-I uses the neighboring samples of previously coded blocks, which are to the left of and/or above the

current block, to determine the prediction error signal, R^l_{PM} ,that corresponds to the difference between the original block, X^l, and the intra prediction signal, P^l_M.

2. When encoding an image block, MMP-I chooses from the available prediction modes the one that achieves the best RD compromise.
3. $R^l_{PM}=X^l-P^l_M$
4. The chosen prediction mode for each image block, is transmitted to the decoder by a prediction mode flag, which is encoded with an adaptive arithmetic coder.
5. This information is encoded prior to the flags and indexes corresponding to the residue block.
6. Once this information has been retrieved, the decoder is able to determine the corresponding prediction block, P^l_M , based on the same previously coded neighboring pixel blocks.
7. It then decodes the approximation of the residue block, R^l_{PM} , and obtains the decoded image block by adding the decoded residue block with the prediction pixels.
8. $X^l= R^l_{PM} +P^l_M$

g. Our-method:

Basic principle lies behind this method is the hierarchical recurrent partitioning of 16*16 blocks which is selected as block size for efficiency. The block is either split horizontally or vertically or no splitting at all. These are represented in the dictionary with the numbers 0, 1, 2 respectively. Two is always followed by its corresponding dictionary index. We have 25 different scales of dictionary and they are 16*16, 16*8, 16*4……1*1. Two is not needed to represent 1*1 sized blocks since it can't be divided further.

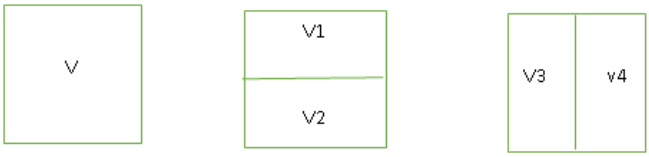

Fig. 3.3: Choices for partitioning.

Figure 3.3 shows the options available for each block for segmentation. It is compared with the corresponding part in the original block to calculate the distortion. If distortion is minimum that block is taken and coded while the particular partitioning is adopted. If minimum lies in V then no partitioning is needed. If v1 is having minimum 0 is added in the dictionary to represent horizontal partitioning, also v1 is coded using dictionary index whereas v2 is further partitioned using same procedure recursively. Similar strategy is used in case of vertical partitioning where minimum lies in either v3 or v4.

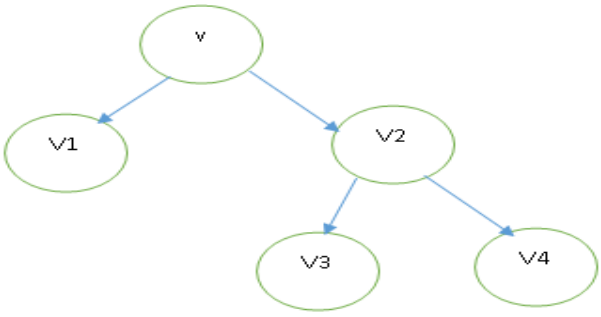

Fig 3.4: partitioning of node V as half tree

In the above given example, if v2 is partitioned further it will be partitioned like showed in Fig.3.5 In the previous versions v1 keep as it is resulting in a half binary tree. In our method v1 is again compared with its corresponding part and distortion is found. If this distortion is greater than a threshold it is portioned again. Thus it becomes a full binary tree as shown in figure 4.6

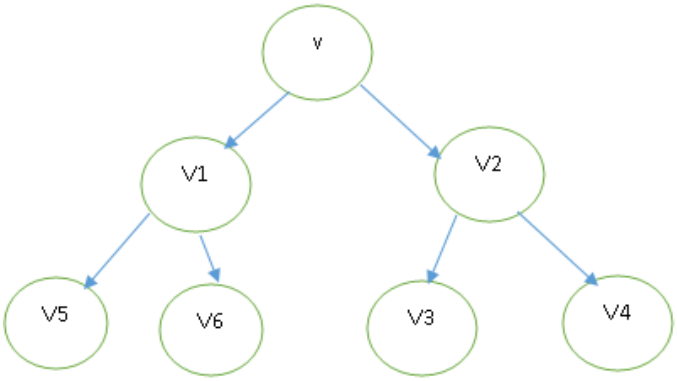

Fig. 3.5: partitioning of node V as full tree

Dictionary is created in both encoding and decoding stages. In previous versions of MMP these were 64 different dictionary of same value uniquely distributed from 0 to 255. In this method this restriction is applied only with those dictionaries whose length is 16 or 8. This restriction is made to avoid dictionary index which may reduce the efficiency of arithmetic encoder. This is avoided for dictionaries having length 1 and 2 and 4 because it won't overcome the barrier 512 even we allow all the 256 variants. Anyhow it need to code all the values below 512 thus it won't reduce coding efficiency. The list of dictionaries for which restriction is made is follows-16*16, 16*8, 8*16, 8*8, 4*16, 4*8, 2*16, 2*8, 1*16, 1*8 (64 blocks) and 16*4, 8*4, 4*4, 2*4, 1*4 (128 blocks).

Execution time is the main overhead of this coding technique. Dictionary learns from new patterns found in the image, as a result dictionary size increases rapidly. So it may take hours to code large images. We can see the result in graphs drawn by taking PSNR vs. bpp. PSNR raises hastily compared to other methods after a certain bpp. Therefore time complexity is exponential with size of image. It can be reduced up to an extent by implementing binary search when selecting the best block from dictionary. The initial dictionary is created in the increasing order therefore binary searching is followed in this range. Linear searching is proposed after this range since it is not sequential.

This method uses Lagrangian optimization, J (T) =R (T) + χD(T). Even though not exactly the same, here it uses a simplified version of that. For every five variants distortions are multiplied with lambda value. So if lambda is greater than one, greater the chance for partitioning since it compares distortion with the constant threshold. Subsequently as lambda value increases, size of compressed file also increases with less distortion. Lambda=1 plays as a value which compromises both size and distortion rates.

The distortion method calculates average difference with respect to a pixel. The block found in dictionary is compared with its original block part, then their difference is calculated for each pixel value. Then applying absolute function followed by summation of their value. It is divided by the size of block resulting in the average value.
MAT=sum (sum (abs (MAT1-MAT2))/m*n;

Result analysis:

Result analysis is based on two functions bpp and PSNR. These are graphs for three different classes of images by taking bpp on X axis and PSNR on Y axis.

In computer graphics, color depth or bit depth is the number of bits used to indicate the color of a single pixel in a bitmapped image or video frame buffer. This concept is usually quantified as bits per pixel (bpp), which specifies the number of bits used. Color depth is only one aspect of color representation, expressing how finely levels of color can be expressed (a.k.a. color precision). The other aspect is how broad a range of colors can be expressed (the gamut). The definition of both color precision and gamut is accomplished with a color encoding specification which assigns a digital code value to a location in a color space.

The phrase peak Signal-to-Noise Ratio, often abbreviated PSNR, is an engineering term for the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. Because many signals have a very wide dynamic range, PSNR is usually expressed in terms of the logarithmic decibel scale.

The PSNR is most commonly used as a measure of quality of reconstruction of lossy compression codecs (e.g., for image compression). The signal in this case is the original data, and the noise is the error

introduced by compression. When comparing compression codecs it is used as an approximation to human perception of reconstruction quality, therefore in some cases one reconstruction may appear to be closer to the original than another, even though it has a lower PSNR (a higher PSNR would normally indicate that the reconstruction is of higher quality). One has to be extremely careful with the range of validity of this metric; it is only conclusively valid when it is used to compare results from the same codec (or codec type) and same content.

It is most easily defined via the mean squared error (*MSE*). Given a noise-free *m*×*n* monochrome image *I* and its noisy approximation *K*, *MSE* is defined as:

$$MSE = \frac{1}{m\,n}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2$$

The PSNR is defined as:

$$\begin{aligned} PSNR &= 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right) \\ &= 20 \cdot \log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right) \\ &= 20 \cdot \log_{10}(MAX_I) - 10 \cdot \log_{10}(MSE) \end{aligned}$$

Here, MAX_I is the maximum possible pixel value of the image. When the pixels are represented using 8 bits per sample, this is 255. More generally, when samples are represented using linear PCM with *B* bits per sample, MAX_I is 2^B-1. For color images with three RGB values per pixel, the definition of PSNR is the same except the MSE is the sum over all squared value differences divided by image size and by three. Alternately, for color images the image is converted to a different color space and PSNR is reported against each channel of that color space, e.g., YCbCr or HSL.

Typical values for the PSNR in lossy image and video compression are between 30 and 50 dB, where higher is better. Acceptable values for wireless transmission quality loss are considered to be about 20 dB to 25 dB.

Here are different images representing images obtained by using various encoding techniques. Figure 4.1 is the original scan_0001.jpg. Compressed image using our method is shown in figure 4.8.

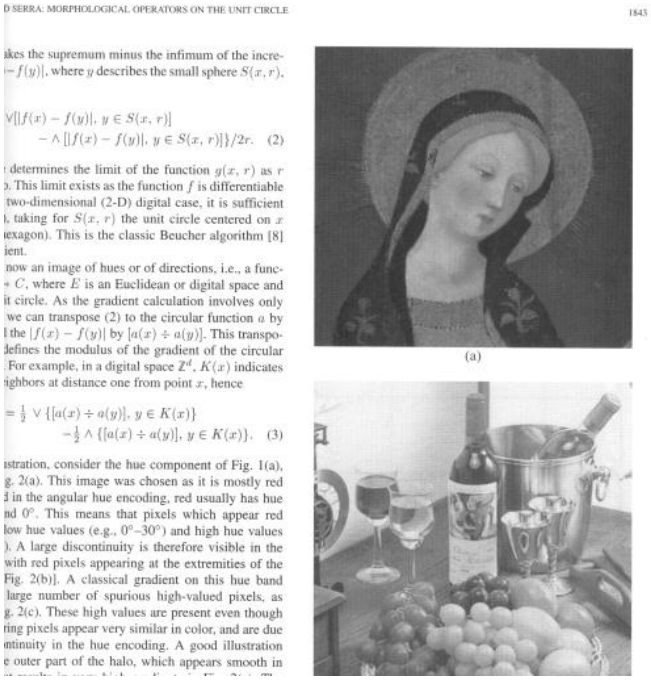

Fig. 5.13: scan_001.pgm

Figure 4.2 represents scan_001 encoded using DjVu. Figure 4.3 is the H.264 encoded version. And JPEG-2000 encoded image is shown in figure 4.4. MMP encoded image is shown in figure 4.5.

a. MMP-FP:

The MMP-FP algorithm was used to encode smooth grayscale images, initially divided into blocks of 16 X 16 pixels, with the prediction segmentation only defined for blocks with more than 16 pixels. This option was a trade-off between compression efficiency and computational complexity. The dictionary optimization parameters used for MMP-II were maintained for MMP-FP. In

Fig.5.18, the results of MMP-FP are compared with those of the previous version of the algorithm (MMP-II) and with those of the original MMP algorithm, for the smooth images Lena, as well as with the referred state-of-the-art encoders.

Fig. 4.2: DjVu encoded scan_001.pgm

Fig. 4.3: H264 encoded scan_001.pgm

Fig 4.4: JPEG/2000 encoded scan_001.jpg

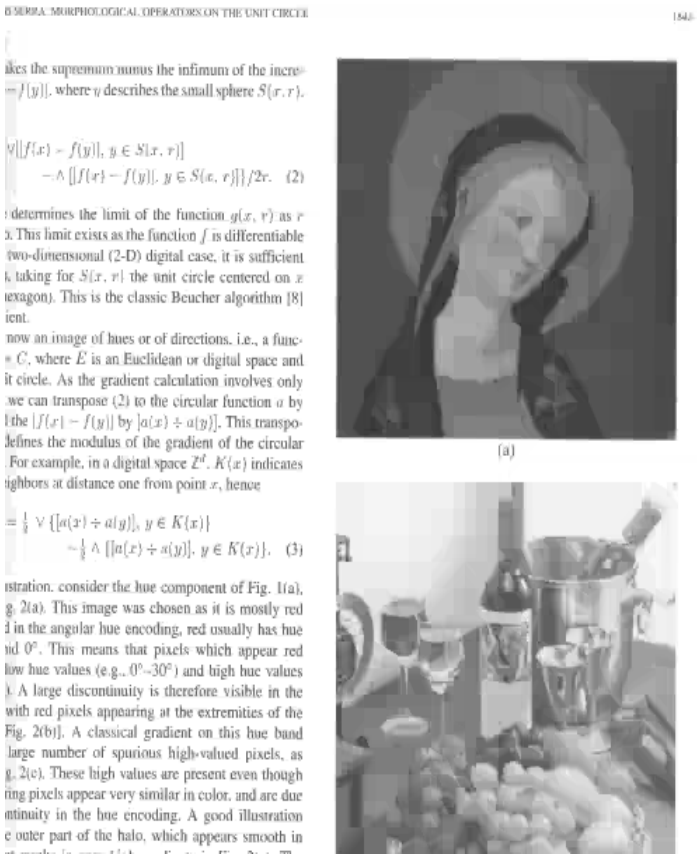

Fig. 4.5: MMP encoded scan_001.pgm.

Fig. 4.8: Experimental results for SCAN002 (512*512).

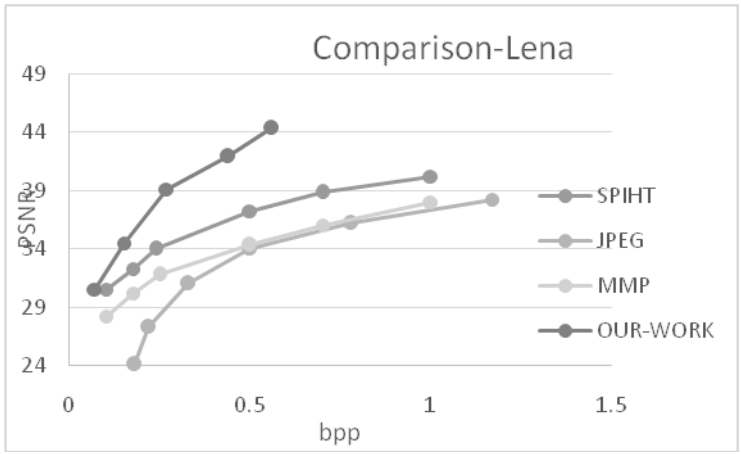

Fig. 4.5: Experimental results for smooth images LENA

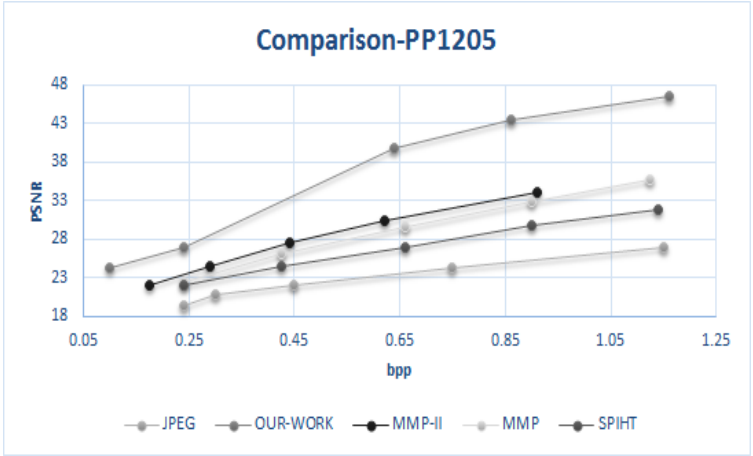

Fig. 4.6: Experimental results for text, image PP1205

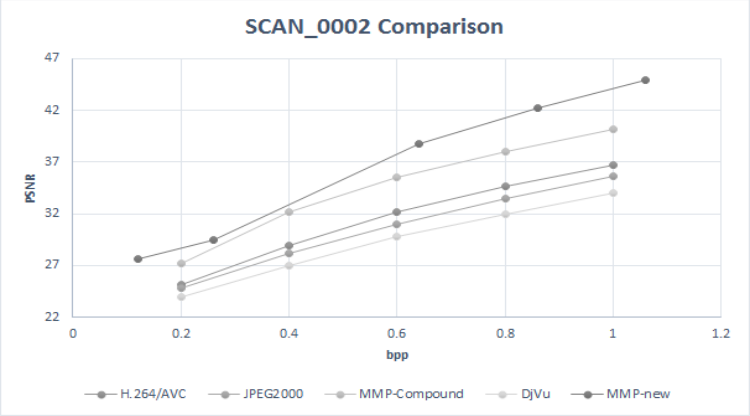

Fig. 4.7: Experimental results for SCAN002 (512*512).

Conclusion and future work:

The new algorithm uses a block classification approach, decomposing the image into smooth and nonsmooth regions. Different MMP-based encoders (MMP-FP and MMP-text) were specifically optimized for each image type. MMP-FP introduces a flexible segmentation scheme that is able to exploit the images' structure in a more efficient way, allowing this method to outperform state-of-the-art DWT and DCT-based encoders for smooth images.

Prediction methods and segmentation used in the process produces only negligible impact in the coding efficiency. Except redundancy control every other techniques used in the dictionary adaptation produce only slight effect. So this method is simple and can stand independently with respect to the older versions of MMP.

From the experimental results itself, it is revealed that it can be used as a dominant method over existing ones for scanned document compression. Execution time is the main problem now facing. More results can be obtained by doing further studies in this area.

REFERENCES

de Carvalho, M., E. da Silva and W. Finamore, 2002. "Multidimensional signal compression using multiscale recurrent patterns," Signal Process, 82: 1559–1580.

Haffner, P., L. Bottou, P.G. Howard and Y.L. Cun, 1999. "Djvu : Analyzing and compressing scanned documents for internet distribution," in Proc. Int. Conf. Document Analysis and Recognition, pp: 625–628.

Haffner, P., L. Bottou, P.G. Howard and Y.L. Cun, 1999. "DjVu: Analyzing and compressing scanned documents for internet distribution," in Proc. Int. Conf. Document Analysis and Recognition, pp: 625–628.

Huttenlocher, D., P. Felzenszwalb and W. Rucklidge, 1999. "DigiPaper: A versatile color document image representation," in Proc. IEEE Int. Conf. Image Processing, Kobe, Japan, pp: 219–223.

ITU-T Recommendation T.44, 1998. Mixed Raster Content (MRC), StudyGroup-8 Contribution.

Marpe, D., T. Wiegand and S. Gordon, 2005. "H.264/MPEG4-AVC fidelity range extensions: Tools, profiles, performance, and application areas" in Proc. IEEE Int. Conf. Image Processing, 1: 593–596.

Nelson Francisco, C., M.M. Nuno Rodrigues, 2010. "Scanned Compound Document Encoding Using Multiscale Recurrent Patterns", IEEE Trans. on Image Processing, VOL. 19, NO. 10.

Optimization and implementation of the integer wavelet transform for image coding IEEE Transactions on image processing vol 11.no.6.June 2002.

Pennebaker, W. and J. Mitchel, 1993. JPEG: Still Image Data Compression Standard. New York: Van Nostrand.

Rodrigues, N.M.M., E.A.B. de Silva, M.B. de Carvalho, S.M.M. de Faria and V.M.M. Silva, 2008. "On dictionary adaptation for recurrent pattern image coding," IEEE Trans. Image Process, 17(9): 1640–1653.

Said, A. and W.A. Pearlman, 1996. "A new fast and efficient image codec based on set partitioning in hierarchical trees," IEEE Trans. Circuits Syst. Video Technol, 6(3): 243–250.

Text Segmentation for MRC Document Compression IEEE Transactions on image processing, vol. 20, no. 6, June 2011.

Text segmentation for MRC document compression IEEE transactions on image processing, vol. 20, no. 6, June 2011.

The JPEG Still Picture Compression Standard Gregory K. Wallace Multimedia Engineering Digital Equipment Corporation Maynard, Massachusetts Submitted in December 1991 for publication in IEEE Transactions on Consumer Electronics

Zaghetto and R. L. de Queiroz, "Iterative pre- and post-processing for MRC layers of scanned documents," in Proc. IEEE Int. Conf. Image Processing, CA, Oct. 2008, pp. 1009–1012.

Zaghetto, A. and R.L. de Queiroz, 2007. "Segmentation-driven compound document coding

based on H.264/AVC-Intra," IEEE Trans. Image Process, 16(7): 1755–1760.

Zaghetto, A. and R.L. de Queiroz, 2008. "Iterative pre- and post-processing for MRC layers of scanned documents," in Proc. IEEE Int. Conf. Image Processing, CA, Oct., pp: 1009–1012.