A Method for Lossless Compression of Images

Zlatoliliya Ilcheva, Valeri Ilchev

Abstract: *Compression of Images is one of the most actual and dynamic developing fields of Informatics and Informational Technologies.*

The suggested method for lossless compression of images requires that the image is presented as a set of 256 image levels, which must contain pixels from the initial image with identical intensity values.

An algorithm containing 4 stages is applied to each level. Each stage of the algorithm is included depending on the satisfying of a criterion for closeness between the pixels of the level.

The method can work as lossless or as perceptually lossless depending on its settings.

The method is universal and gives good results comparable to JPEG regarding images of different types and is open for further development at all stages.

Key words: *Lossy and lossless compression, image function, image level, criterion for pixel closeness, straight and inverse cos-transformation.*

INTRODUCTION

The paper is connected with one of the most actual and dynamic developing fields of **Informatics** and **Informational Technologies.**

The image processing is already an integral part of the computer systems which work in a wide range of human activities like industry, medicine, banking, ecology, cartography, book publishing, geology, transport, criminalistics, military, aviation and space industry [1, 2, 4, 5].

Computer images have one essential disadvantage – they need a huge amount of memory. This fact creates problems when image files are exchanged as well as when libraries of images are created and worked with.

The Internet offers a comfortable access to information from every point of the world but the capacity of the connection is too limited. Most users still use a standard telephone line and a 56 Kbps modem. The need for powerful image compression methods, which work fast enough on the contemporary hardware and preserve the quality of the compressed images is more sharply-felt than ever.

DESCRIPTION

Two types of methods for image compression exist – **lossy** and **lossless** methods. Each of them has its preferred fields of application [1,3,7].

The goal of **lossless** image compression is to represent an image signal with the smallest possible number of bits without loss of ***any*** information. The decompressed image data should be identical both quantitatively (numerically) and qualitatively (visually) to the original compressed image. The methods of this type usually give compression factor between 1.5 and 3.

In order to achieve higher compression factors, the so called **perceptually lossless**[1] compression methods attempt to remove redundant as well as perceptually irrelevant information; these methods require that the compressed and decompressed images be only visually, and not necessarily numerically, identical. In this case some loss of information is allowed as long as the recovered image is perceived to be identical to the original one.

The method presented in this article may generally be considered as **lossless**. Depending on the requirements of the concrete application it can be set to work as **perceptually lossless (**and even as **lossy**)**.** It is necessary to point out that in this paper only the main elements of the encoder will be discussed since the decoder performs the inverse operations of the encoder.

Let **z=f(x,y)** is a function of the image, defined in the three-dimensional Euclidean space. Every pixel of the image is specified in this space by the triad **(x,y,z),** where **x** and **y** are the coordinates of the pixel in the plane **(x,y)** and **z** is the intensity of the pixel. So presented, in the three-dimensional space the image is *a surface* in which the height of each pixel **z** is a number in the range (0,255).(As known, the number of the gray levels in the images is 256). In case of color image it is worked separately and identically on the three channels of the color model.

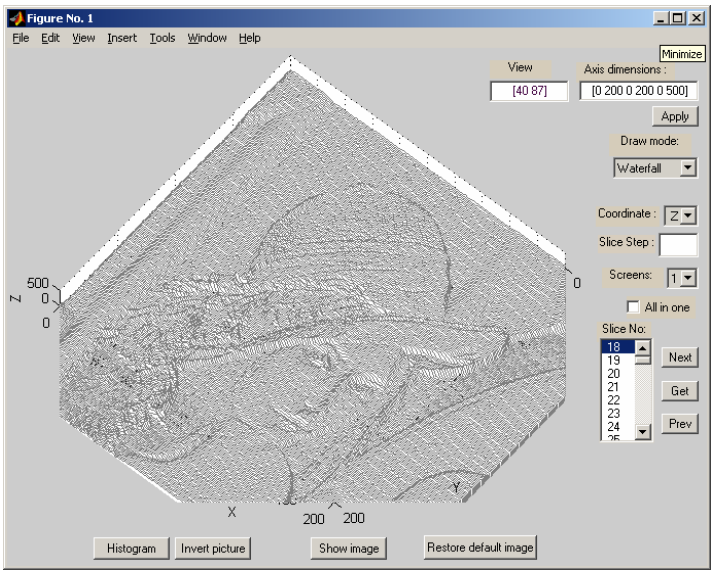


Fig. 1

Let us assume that we have passed planes, parallel to the plane **(x,y)** and perpendicular to the axis **z** of each of the levels of **z** $(0 \le z \le 255)$. Thus we obtain 256 two-dimensional images and each of them contains pixels of the initial image, which have one and the same intensity. In fact it is a way of image coding which is identical to the raster coding. The difference is that in raster coding the intensities of the pixels following in raster order are saved, while in the suggested method arrays containing coordinates of the pixels with constant intensity are saved.

We introduce the term **image level z** $(0 \le z \le 255)$ as a set of image pixels with identical intensity. Each level of the image is processed separately with suggested method. The advantage is that the level contains information of **black & white** type and bit by bit representation of the data may be looked for.

All levels of the image are obtained with the program. Each level is divided into areas of size 256x256 pixels. Algorithm is applied on each of these areas separately. At the beginning of the level record of 5 bytes is stored, indicating presence or absence of image pixels in these biggest areas of the level. The number of these bytes may vary depending on the size of the images. The bits in these bytes are formed in the run of the algorithm.

On each area 256x256, a raster movement is performed, usually using a 16x16 matrix containing 16 inclusion matrixes 4x4. Each level is processed using an algorithm which contains 4 stages. Each stage is turned on, depending on the closeness between the pixels of the level. It begins with looking for and compressing pixels of biggest degree of closeness (stage 1) and ends with looking for and compressing pixels of smallest degree of closeness (stage 4). The term **degree of closeness in the discrete plane of the pixels** is formulated very generally as a presence of a definite minimum number of image pixels in the 4x4 matrix, without taking into account their location in this matrix and as a presence of a definite minimum number of pixels in the 16x16 matrix.

Stage 1.

At this stage the so called **runs** are searched in every level of the image. We regard as a run a horizontal line of 3 or more pixels following one after the other (3 or more horizontal pixels with a zero space between them) [7]. The structure of the record after this stage is as follows:

stage3 bytes for each run – one byte for the **x** and one for the **y** coordinate of the first pixel of the run and one byte for the number of pixels in the run.

The pixels, encoded at this stage are deleted from the image of the level. The second stage of the algorithm follows.

Stage 2.

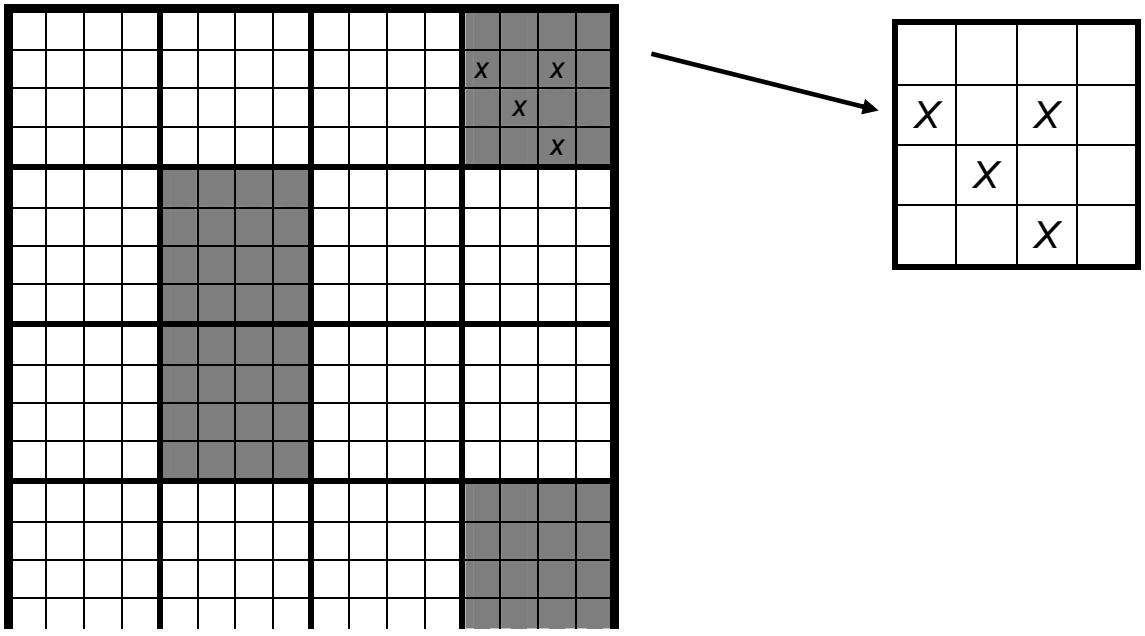
At this stage well grouped pixels (pixels in compact groups) are searched from the area of the image.

In the 256x256 area a raster scanning using 16x16 matrix is performed. This matrix contains 16 sub-matrixes 4x4. At this stage the criterion for closeness between the pixels , i. e. the criterion for its including in the compressing algorithm, is that the matrix 16x16 should contain at least 3 sub-matrixes 4x4 with sufficient number of image pixels. In our case this sufficient number is 4 pixels.

Considering this, the encoding of the image level is performed in the following way (Fig. 2):

2 bytes for the address of the upper left pixel of the 16x16 matrix – one for the **x** and one for the **y** coordinate. 2 bytes describing the inner structure for this block follow. This structure reflects the position of the 4x4 sub-matrixes which contain 4 or more image pixels. Two bytes for each 4x4 sub-matrix if this kind follow successively. The other matrixes 4x4 which do not satisfy this condition are not encoded at this.

(X, Y)



2 bytes		2 bytes	2 bytes	2 bytes
X	**Y**	0 0 0 1 0 1 0 0 0 1 0 0 0 0 0 1	0 0 0 0 1 0 1 0 0 1 0 0 0 0 1 0	. . . . .

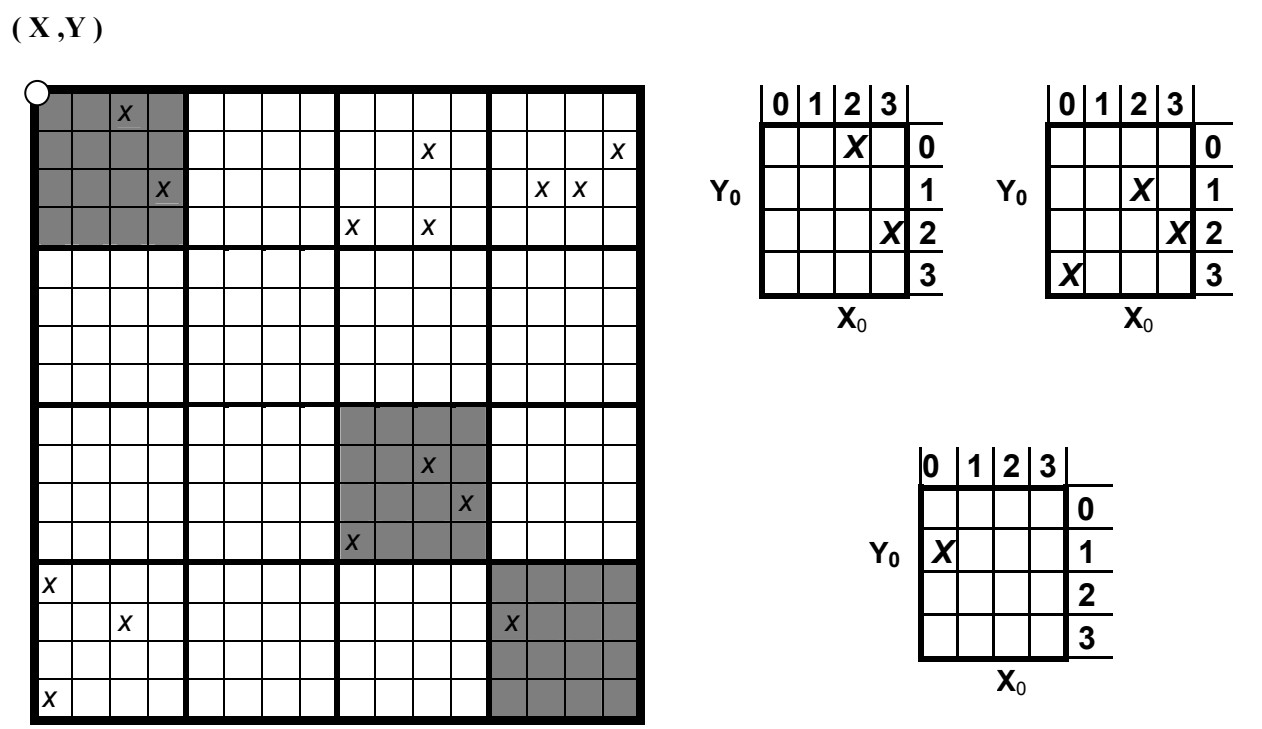
Fig. 2

The encoded pixels at this stage are also deleted from the initial image.

Stage 3.

At this stage the pixels with more smaller degree of closeness than these of the previous stages, the so called "clouds of pixels " are processed. The pixels are far one from another, but still satisfy the condition for compactness [2].

Here, again in the 256x256 area a raster scanning is performed using a 16x16 matrix , which contains 16 sub-matrixes 4x4. The encoding of the image area is performed in the following way: (Fig. 3)



2 bytes		4 bytes														4 X 4				4 X 4								4 X 4				
X	Y	1	0	1	1	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	1	1	1	0	1	0	0	1	1	1	1	0	0	0	1	1	0	0	0	1	...
		0	0	1	1	0	0	0	0	0	0	1	0	1	0	0	1	X_0	Y_0	X_0	Y_0	X_0	Y_0	X_0	Y_0	X_0	Y_0	X_0	Y_0	X_0	Y_0										

Fig. 3

2 bytes for an address of the upper left corner of the 16x16 matrix. 4 bytes for the structure of this matrix follow. In a pair of bits in these 4 bytes in raster order the number image pixels in the 4x4 matrixes is saved. Here (0,0) means 0 pixels, (0,1) – 1 pixel, (1,0) – 2 pixels, (1,1) – 3 pixels. The records of the relative coordinates of the pixels in the 4x4 sub-matrixes follow and we have for $\mathbf{X_0}$ and $\mathbf{Y_0}$ of each pixel two bits, i. e. $\mathbf{X_0}$ and $\mathbf{Y_0}$ are relative addresses of the pixels in the 4x4 matrix. Each point is saved in one semi-byte and the records follow sequentially without any separators. This relative address registration of the pixels is possible due to the fact that the topological information for them is already encoded in the 4 structural bytes at the beginning of the record. Compression is evidently present when there are more than12 pixels in the 16x16 matrix. The algorithm takes care for this number and when the condition for presence of at least 12 image pixels is not satisfied, the pixels in this area are processed at the next stage.

It is necessary to note that in the 4x4 matrixes at stage 3 is not possible to have more than 3 pixels since these matrixes are already encoded at the previous stage 2, i. e. there is no condition for grouping of pixels in the 4x4 matrixes.

The pixels encoded at stage 3 of the algorithm are deleted from the processed area.

Stage 4.

After the three stages of the algorithm in each of the image levels single pixels with a higher degree of remoteness one from another remain. The single pixels from each levels are projected again on the plane **(x,y)** each with the value of its intensity, i. e. the rest-image, consisting only of single distant pixels is processed as a matrix of numbers in the range (0,255). These pixels are arranged in an area with a structure byte per pixel, following raster one after the other and without separators between them.

A **cos**-transformation is performed on these pixels. From theory is known [2,5] that the straight **cos**-transformation of the function I(n) is carried out by the following formula:

$$b_k = \frac{2}{N}\sum_{n=1}^{N}[I(n).\cos(\frac{\pi kn}{N})],$$

where: **k=** 0,1,2,…, **K**;

k is the number of the transformation coefficients;

K- count of coefficients;

n –number of the pixels;

N – count of all pixels.

This transformation is carried out in the encoder.

The obtained coefficients are saved sequentially in an array and the coefficients with small values (i. e. the frequencies with small values of the amplitudes in the cos-transformation) are nullified. In our case, the empirically specified threshold **t,** under which the coefficients are nullified is 0,05. After the saving of the topology of the coefficients' array, it is shortened by removing the null records.

The reverse **cos**-transformation is carried out in the decoder using the formula:

$$I(n) = \frac{b_0}{2} + \sum_{k=1}^{K}[b_k.\cos(\frac{\pi kn}{N})],$$

where $b_k, k = 0,1,2,...,K$ are the transformation coefficients

The restoring of the compressed pixels of the image can be **lossless** or **perceptually lossless** depending on the choice of the threshold parameter **t**.

CONCLUSIONS AND FUTURE WORK

Initially the method was programmed in MATLAB 6.0 (Math Works, Inc., USA)[6] environment and a vast number of experiments on test and author images of different kinds were carried out. After a good work and universality was achieved at all stages it was programmed in Visual Basic 6.0.

Some compression results are presented on Table 1(gray level test images) and Table 2 (black & white images). A comparison with (.jp2) file format is made.(The used program was Lura Wave Smart Compress 3.0).Here (.mle) is our file format and C is compression coefficient.

Table 1

Image	(.bmp) file [KB]	(.jp2) file		(.mle) file	
		[KB]	C	[KB}	C
lena	66.616	38.754	1.72	44.042	1.51
barbara	263.224	152.756	1.72	149.362	1.76
baboon	263.224	197.819	1.33	121.505	2.17
boat	263.224	159.930	1.65	147.242	1.79
goldhill	263.224	158.807	1.66	124.226	2.12
peppers	263.224	143.303	1.84	155.621	1.69

Table 2

Image	(.bmp) file [KB]	(.jp2) file		(.mle) file	
		[KB]	C	[KB]	C
bw1	8.486	8.902	0.95	0.981	8.65
bw2	59.550	68.933	0.85	27.410	2.17
bw3	11.862	14.552	0.82	3.982	2.98
text	662.968	882.082	0.75	393.075	1.69

The large number of experiments allow us to make the following more important conclusions:

1.An important advantage of the method is its universality – it gives a good compression on text and **black & white** graphical images as well as on photo-images (gray-level images) and color images.

2. The method is especially good at images with many run-s (the text and graphic images) as well as at gray level images with a concentration of pixels on some gray-levels (with picks in the histogram). The method gives here better results than the traditional methods.

3. The method gives good possibilities for future development of some of the coding stages. Till the last stage it is completely **lossless** and it can be worked further on the criteria of closeness between the pixels in the second and the third stage.

4. With the including of the last stage of the method – cos-transformation – the method increases significantly its possibilities regarding the degree of compression. Depending on the threshold parameter **t,** it could work as **lossless** or **perceptually lossless**. Naturally, the degree of compression is very different for these modes.

5. It is interesting to carry out a research how this method would work when level on the **z** axis are united in images of the **black & white** type. A source of information for the number of the levels, which are to be united, may possibly be the image histogram. The levels may be united uniformly or non-uniformly in the different parts of the interval (0, 255). These are some of the further work directions on the method.

REFERENCES

[1] *Handbook of Image and Video Processing*. (Ed. Al Bovik), Academic Press, 2000.

[2] Jain A. K. *Fundamentals of Digital Image Processing*. Prentice-Hall. Int.Ed. Englewood Cliffs, New York,1989.

[3] Moffat A., T. C. Bell, I. H. Witten. *Lossless Compression for Text and Images*. University of Melbourne Australia, Dep. Of Comp.Science, Facsimile, Oct.1995.

[4] Parker J. R. *Algorithms for Image Processing and Computer Vision*. John Wiley&Sons,Inc., 1997.

[5] Pratt W., K. *Digital Image Processing*. John Wiley& Sons, 1978.

[6] Redfern D., C. Campbell. *The MATLAB5 Handbook*. Springer-Verlag New York, Inc., 1998.

[7] Salomon D. *Data Compression*. Springer, 1997.

ABOUT THE AUTHORS

Assoc. Prof. Zlatoliliya Ilcheva, PhD, Institute of Computer and Communication Systems, Bulgarian Academy of Sciences, Phone: (+359 2) 73 29 51 (ext 119), E-mail: zlat@agatha.iac.bg

MSc. Valeri Ilchev, Institute of Information Technologies, Bulgarian Academy of Sciences, Phone: (+359 2)70 75 86, E-mail: ilch@iinf.bas.bg