



DISCRETE COSINE TRANSFORM BASED IMAGE COMPRESSION

Aniket S. Dhavale¹, Ganesh B. Gadekar², Mahesh S. Bhagat³, Vitthal B. Jagtap⁴

^{1,2,3,4}SBPCOE Indapur, S P University of Pune, Maharashtra

Email: aniket2727@gamil.com¹, 2727ganesh@gamil.com², bhagatmahesh33@gamil.com³, vilsjagtap@gamil.com⁴

Abstract

Now a days image compression plays a important role in image storage and image transformation. When we get the number of bits per image from sampling rates and quantization methods, we knows that image compression is required. Image compression is one of the type of data compression. Image compression have mainly two types lossy or lossless. In this paper using only MATLAB functions it is being attempted to implement JPEG compression. In this lossy compression techniques is used, where data loss cannot affect the image quality. It is a survey for lossy image compression using DCT transform, it contains JPEG compression algorithm and describes all the components of it.

Keywords: Image Compression, JPEG, DCT, Quantization.

I. INTRODUCTION

Compressing an image is slightly different than compressing raw binary data. To compress images, general purpose compression programs can be used. Discrete cosine transform (DCT) [1,2] is the basis of many image compression methods. For example, the standard JPEG [3,4], for which DCT is carried out in 8x8 image blocks existed as the main image compression standard for about 10 years. However, many investigations and achieved progress in this area have dealt with applications of discrete wavelet transform (DWT). For example, the compression standard JPEG2000 [5,6] accepted quite recently is based on DWT and it commonly provides

considerably better quality of decoded images than JPEG. Aforesaid allows supposing that DWT is more appropriate transform for applying in image compression than DCT. In this paper we try to show that this is not true. Due to rather simple improvements of the base method (used in JPEG) it is possible to obtain decoded images quality better than for JPEG2000. There are three basic modifications introduced by us compared to JPEG. First, an image is divided into 32x32 pixel blocks instead of 8x8 for conventional JPEG. Second, the quantized DCT coefficients are divided into bit-planes; the bit values are coded according to complex probability models that take into account the presence of correlation between values of neighbor coefficients in blocks and between the values of the corresponding coefficients of neighbor blocks. Third, DCT based filtering [7] is used as post-processing for removal of blocking artifacts from decoded images and, thus, for increasing decoded image quality.

1.1 Introduction to Transformation:

Transform coding constitutes an integral component of contemporary image/video processing applications. Transform coding relies on the premise that pixels in an image exhibit a certain level of correlation with their neighboring pixels. Similarly in a video transmission system, adjacent pixels in consecutive frames show very high correlation. Consequently, these correlations can be exploited to predict the value of a pixel from its respective neighbors. A transformation is, therefore, defined to map this

spatial (correlated) data into transformed (uncorrelated) coefficients. Clearly, the transformation should utilize the fact that the information content of an individual pixel is relatively small i.e., to a large extent visual contribution of a pixel can be predicted using its neighbors. A typical image/video transmission system is outlined in Figure 1. The objective of the source encoder is to exploit the redundancies in image data to provide compression. In other words, the source encoder reduces the entropy, which in our case means decrease in the average number of bits required to represent the image. On the contrary, the channel encoder adds redundancy to the output of the source encoder in order to enhance the reliability of the transmission. In the source encoder exploits some redundancy in the image data in order to achieve better compression. The transformation sub-block de-correlates the image data thereby reducing inter pixel redundancy. The transformation is a lossless operation, therefore, the inverse transformation renders a perfect reconstruction of the original image. The quantize sub-block utilizes the fact that the human eye is unable to perceive some visual information in an image. Such information is deemed redundant and can be discarded without introducing noticeable visual artifacts.

1.2. Mean Square Error (MSE):

Mean square error is a criterion for an estimator: the choice is the one that minimizes the sum of squared errors due to bias and due to variance. The average of the square of the difference between the desired response and the actual system output. As a loss function, MSE is called squared error loss. MSE measures the average of the square of the "error". The MSE is the second moment (about the origin) of the error, and thus incorporates both the variance of the estimator and its bias. For an unbiased estimator, the MSE is the variance. In an analogy to standard deviation, taking the square root of MSE yields the root mean squared error or RMSE. Which has the same units as the quantity being estimated. for an unbiased estimator, the RMSE is the square root of the variance, known as the standard error.

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

Where $m \times n$ is the image size and $I(i, j)$ is the

input image and $K(i, j)$ is the retrieved image.

1.3. Peak Signal-to-Noise Ratio(PSNR):

It is the the ratio between the maximum possible power of a signal and the power of corrupting noise .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 in image compression etc. It is most easily defined via the mean squared error (MSE) which for two $m \times n$ monochrome images I and K where one of the images is considered noisy.

The PSNR is defined as:

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

II. COADING AND DECODING SCHEMES

The block-diagram of image coding for the proposed approach is depicted in Fig. 1. Image DCT 32x32 Quantization of DCT coefficients Dividing into bitplanes Classification and coding of bit values of bit-planes Compressed image.

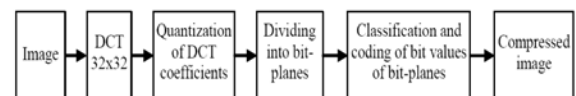


Fig. 1. The block-diagram of image coding

An image to be compressed is divided into 32x32 pixel blocks. Then, DCT for pixel values of each block is computed. After this, the quantization of DCT coefficients of image blocks is carried out. At this stage the basic losses are introduced into compressed image. Larger quantization step (QS) provides larger compression ratio (CR) and simultaneously it leads to larger losses. In this paper it is proposed to use uniform quantization that ensures the best results within the structure of the considered method.

Then, the division of quantized DCT coefficients into bit-planes is carried out. The obtained bit-planes are coded in the order starting from higher bits to lower ones. While coding each next plane,

the values of bits of earlier coded planes are taken into account. A coded bit is referred to one or another group of bits according to the values of already coded bits. For each group of bits, individual probability model is used for dynamic arithmetic coding (see Section 3). The block-diagram of image decoding is presented in Fig. 2 where IDCT denotes inverse DCT. IDCT Image 32x32 Deblocking Merger of bit planes Classification and decoding of bit values of bit-planes Compressed image.

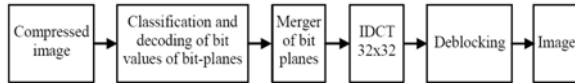


Fig. 2. The block-diagram of image decoding.

As seen, at image decoding stage all steps are repeated in reverse order. Besides, at the final step the operation of decoded image filtering is added (see Section 4 for more details).

2.1 Discrete Cosine Transform (DCT):

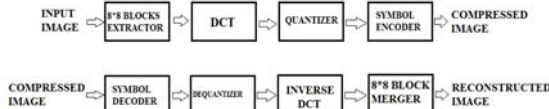


Fig.3 Image Compression using DCT

The discrete cosine transform (DCT) is a technique for converting a signal into elementary frequency components. Like other transforms, the Discrete Cosine Transform (DCT) attempts to de correlate the image data. After de correlation each transform coefficient can be encoded independently without losing compression efficiency.

2.2 Proposed DCT Algorithm:

- The following is a general overview of the JPEG process. The image is broken into 8x8 blocks of pixels.
- Working from left to right, top to bottom, the DCT is applied to each block.
- Each block is compressed through quantization.
- The array of compressed blocks that constitute the image is stored in a drastically reduced amount of space.
- When desired, the image is reconstructed through decompression, a process that

uses the inverse Discrete Cosine Transform (IDCT).

III. BIT-PLANE COADING

Thus, after calculation of DCT in 32x32 blocks and quantization of obtained coefficients, we have an array of integer valued DCT coefficients. Divide the array of absolute Values of DCT coefficients into n bit-planes, where n is the number of the highest bit-plane in which there are non-zero values. Coding begins with the bit-plane n and comes to an end by bit-plane 1. The signs of non-zero DCT coefficients are practically random variables with approximately equal probabilities. Therefore, they are allocated into a separate array (one bit for each sign) and transferred to the output stream at once. Let $P_{k,l,m}(i,j)$ defines a bit value of a bit-plane k of a coefficient with the index i,j of the block of an image with the index l, m, where $k=1..n, i,j=1..32, l=1..L, m=1..M$, L,M denotes the number of image blocks for vertical and horizontal directions. We introduce the following conditions which are used for classification of bits of bit-planes:

- 1) $C1(k,l,m,i,j)=true$, if $1 \in \{P_{k+1,l,m}(i,j), \dots, P_{n,l,m}(i,j)\}$. This condition is assigned true if, at least, one bit among earlier coded higher bit planes is equal to 1.
- 2) $C2(k,l,m,i,j)=true$, if $1 \in \{P_{k+2l,m}(i,j), \dots, P_{n,l,m}(i,j)\}$. This condition is true if, without taking into account the previously coded higher bit-plane, the bit with these indices was equal to 1. If the condition C2 is true then the current bit with approximately equal probability can be equal either to 0 or to 1. If the condition C1 is true and the condition C2 is false then the probability of zero for the current bit is considerably larger than the probability to be equal to 1.
- 3) $C3(k,l,m,i,j)=true$, if $1 \in \{P_{k,l,m}(i,j), \dots, P_{n,l,m}(i,j)\}$. This condition is true if in this or in, at least, one of earlier coded higher bit planes the bit with these indices was equal to 1. The condition C3 can be checked for those bits neighboring the coded bit that till the current moment have been already coded. Here and below only the values of those bits can be checked that have been already coded. This is important for providing an opportunity of decoding. At decoding stage those bits that have

been coded earlier are decoded earlier as well and they can be checked in conditions.

4) $C4(k,l,m,i,j)=true$, if $P_{k+1,l,m}(i,j)=1$. This condition is true if in the previously coded bit plane the bit with these indices was equal to 1.

5) $C5(k,l,m,i,j)=true$, if $P_{k,l,m}(i,j)=1$.

6) $C6(k,l,m,i,j)=true$, if $true \in \{C1(k,l,m,i-1,j-1), C1(k,l,m,i-1,j), C1(k,l,m,i-1,j+1), C1(k,l,m,i,j-1), C1(k,l,m,i,j+1), C1(k,l,m,i+1,j-1), C1(k,l,m,i+1,j), C1(k,l,m,i+1,j+1)\}$. This condition is true if for, at least, one of neighboring bits there is unity in higher bit planes.

7) $C7(k,l,m,i,j)=true$, if $true \in \{C5(k,l,m,i-1,j-1), C5(k,l,m,i-1,j), C5(k,l,m,i-1,j+1), C5(k,l,m,i,j-1)\}$. This condition is true if, at least, one among neighboring and already coded bits of this bit-plane was equal to 1.

8) $C8(k,l,m,i,j)=true$, if $true \in \{C3(k,l,m,i-2,j-2), C3(k,l,m,i-2,j-1), C3(k,l,m,i-2,j), C3(k,l,m,i-2,j+1), C3(k,l,m,i-1,j-2), C3(k,l,m,i-1,j+2), C3(k,l,m,i,j-2), C3(k,l,m,i,j+2), C3(k,l,m,i+1,j-2), C3(k,l,m,i+1,j+2), C3(k,l,m,i+2,j-2), C3(k,l,m,i+2,j-1), C3(k,l,m,i+2,j), C3(k,l,m,i+2,j+1), C3(k,l,m,i+2,j+2)\}$.

This condition is true if there was, at least, one unity in this or higher bit planes for already coded bits displaced from the coded bit by 2 rows or 2 columns.

9) $C9(k,l,m,i,j)=true$, if $true \in \{C3(k,l,m,i-3,j-3), C3(k,l,m,i-3,j-2), C3(k,l,m,i-3,j-1), C3(k,l,m,i-3,j), C3(k,l,m,i-3,j+1), C3(k,l,m,i-3,j+2), C3(k,l,m,i-3,j+3), C3(k,l,m,i-2,j-3), C3(k,l,m,i-2,j+3), C3(k,l,m,i-1,j-3), C3(k,l,m,i-1,j+3), C3(k,l,m,i,j-3), C3(k,l,m,i,j+3), C3(k,l,m,i+1,j-3), C3(k,l,m,i+1,j+3), C3(k,l,m,i+2,j-3), C3(k,l,m,i+2,j+3), C3(k,l,m,i+3,j-3), C3(k,l,m,i+3,j-2), C3(k,l,m,i+3,j-1), C3(k,l,m,i+3,j), C3(k,l,m,i+3,j+1), C3(k,l,m,i+3,j+2), C3(k,l,m,i+3,j+3)\}$.

This condition is true if there was unity in this or higher bit planes for already coded bits displaced from the coded bit by 3 rows or 3 columns.

10) $C10(k,l,m,i,j)=true$, if $true \in \{C3(k,l-1,m-1,i,j), C3(k,l-1,m,i,j), C3(k,l-1,m+1,i,j), C3(k,l,m-1,i,j), C3(k,l,m+1,i,j), C3(k,l+1,m-1,i,j), C3(k,l+1,m,i,j), C3(k,l+1,m+1,i,j)\}$.

This condition is true if there was unity in this or in higher bit planes for bits in neighbor blocks. This condition allows taking into consideration correlation for bits having identical indices and belonging to image neighbor blocks.

11) $C11(k,l,m,i,j)=true$, if $(C2(k,l,m,i,j) = false)$ and $(C6(k+1,l,m,i,j) = false)$.

The checking of this condition allows classifying more reliably the bit for which in the previously coded bit plane there was unity.

12) $C13(k,l,m,i,j)=true$, if $1 = C12(k,l,m,i-1,j-1) + C12(k,l,m,i-1,j) + C12(k,l,m,i-1,j+1) + C12(k,l,m,i,j-1)$.

13) $C14(k,l,m,i,j)=true$, if $k=1$.

Fig. 4 presents the flowchart of bit value classification by checking the aforementioned conditions (PMX - probability model number X).

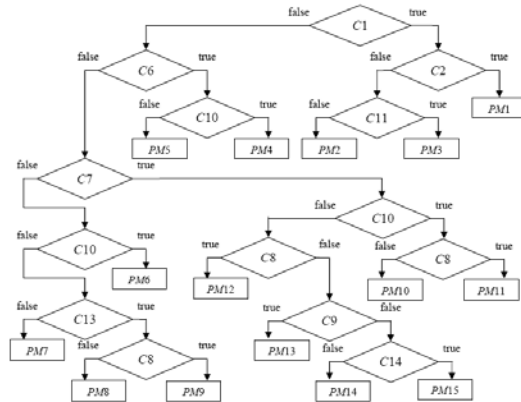


Fig. 4. Flow chart of classification of coded bits of bit-planes

Totally according to given classification a bit can be referred to one of fifteen probability models. For each model after coding the current bit the counters of 0 and 1 are corrected, and they are used for coding next bits referred to this model. For coding it is proposed to use the dynamic version of arithmetic coding [8,9] that is the most effective for the considered case.

Let us consider the obtained classification more in detail. To PM1 and PM2 those bits are referred that for higher bit planes had unities and for which the probabilities of being equal to 0 and 1 are practically equal. To PM3 those bits are referred for which there was unity in previously coded plane and the probability of unity was low. Because of this, for PM3 the probability of 0 is larger than being equal to 1.

To the model PM4 those bits are referred that have no unities in higher bit-planes but there are neighbor bits with unities in higher bit planes and there are unities in the corresponding bits of image neighbor blocks. For PM4 the probabilities of 1 and 0 are rather close, and the bits referred to this model are compressed poorly. The difference between the models PM4 and PM5 consists in the following. To PM5 those

bits are referred that have no unities in the corresponding bits of image neighbor blocks. For the model PM5 there are considerably more zeros than unities, and the corresponding bits are compressed considerably better.

The bits of the models PM6-PM9 differ from the bits of the models PM4 and PM5. For the former ones there are no neighbor bits with unities in higher bit planes, but there are unities for neighbor bits in the current coded plane. For the models PM6- PM9 the probability of unities is considerably smaller than for the models PM4 and PM5. Because of this, these data are compressed better. Those bits are referred to the model PM6 that have unities in the corresponding bits of image neighbor blocks. The bits of this model are compressed in the worst way among the bits that belong to the group of the models PM6-PM9. For the models PM7-PM9 there are no unities in the corresponding bits of image neighbor blocks. For the bits referred to the model PM7 the number of unities in the neighbor bits is larger than 1. For the models PM8 and PM9 there is only one unity in the neighbor bits. Because of this, the probability of unities for them is even smaller than for the model PM7. Division of bits between the models PM8 and PM9 is accomplished using the condition C8 that allows taking into account the presence of unities in the bits displaced from the coded one by 2 rows and 2 columns. Due to this, the bits of the model PM8 for which C8=false are compressed best of all among the bits of the models PM6-PM9.

The bits of the models PM10-PM15 differ from the bits of the models PM4-PM9 by the following. For the former ones there are no unities either in higher bit planes or in the current coded plane. The bits of the models PM10-PM15 are compressed very well, however, additional division of such bits into several models lead to considerable increasing of CR. Those bits are referred to the models PM10, PM11 that have unity in the corresponding bits of image neighbor blocks. The bits of the model PM10 are compressed slightly better since for them there are no unities in the bits displaced from the coded one by 2 rows and 2 columns. For the model PM13, there are unities only in bits displaced from the coded one by 3 rows and 3 columns. For the models PM14 and PM15 there are no unities in the checked area. Such bits are compressed in the best way (most efficiently). The difference

between these models consists in the following. To the model PM15 those bits are referred that belong to the lowest bitplane ($k=1$). We propose to avoid coding the bits of the model PM15 (they all are considered equal to 0). This is analogous to «dead zone» in quantization. But in our case, this occurs to be effective due to selectivity of its application.

Before starting coding each bit plane, the counters of unities and zeros for the models PM1-PM14 are initialized as unities, i.e. the models of each coded plane are independent. Different copies of the models PM1-PM14 are used for different regions of image blocks. For the bits of DCT coefficient with the indices $i=1$, $j=1$ (this is the quantized value of the block mean) a separate copy of the models PM1-PM14 is used. The statistical characteristics of this DCT coefficient considerably differ from statistics of other DCT coefficients. A separate copy of the models PM1-PM14 is also used for the first (upper) row of block DCT coefficients. This is explained by the fact that for these coefficients there is only one earlier coded bit. This leads to considerable difference of bit distribution between the models. For all other DCT coefficients of a block (and they are the basic amount of data) the third copy of the models PM1-PM14 is used.

The proposed classification is obtained by experimental studies of efficiency of various ways to divide bits into classes for different test images and QS. Probably, more effective variant of such classification can be found. In practice, simpler variants of classification can be used in order to increase coding speed. For example, the absence of checking the condition C10 (in this case one does not take into account the correlation between neighbor blocks of an image) results in increasing the size of compressed image by 1-3 %. If one does not check the condition C9 (this condition deals with correlation of bits displaced from the coded one by 3 rows and 3 columns) the size of coded image increases by 1-1.5%. If one also does not check the condition C8 (this condition deals with correlation of bits displaced from the coded one by 2 rows and 2 columns), this leads to the increasing of coded image size by 3-7%.

Let us mention one important point once again. For the used variant that includes the PM15, the losses of image quality occur not only at DCT coefficient quantization step, but also (though in

much smaller degree), at the step of bit values coding for bit planes. If one avoids using the model PM15 this does not lead to any additional losses at this step.

IV. FILTERING FOR REMOVAL OF BLOCKING ARTIFICIAL

For blocking effect reduction in decoded images, we employ an approach described in[7]. This approach presumes the use of DCT based filter for additive noise removal[10]. In the considered case, the noise to be removed is the quantization noise. The size of a sliding window of the DCT based filter is 8x8. For each position of the sliding window, DCT is carried out, then DCT coefficients having absolute values smaller than preset threshold are assigned zero values (hard thresholding). After this, inverse DCT is executed.

Spatially invariant denoising is employed. One problem is the setting of the threshold. For our application we recommend to set the threshold equal to $QS/2$ (note that we know QS a priori). The use of post-filtering in our case allows increasing quality of the decoded images by 0.5-1 dB. The decoding time increases by 30-40 %.

V. NUMERICAL SIMULATIONS

The quality of compression for the proposed method was analyzed for 512x512 grayscale images in comparison to JPEG2000 (Kakadu coder by D.Taubman [6] has been employed). The practical realization of our method in programming language Delphi (the coder has the name AGU) is accessible to downloading from the <http://www.cs.tut.fi/~karen/aguCoder.htm> address. This version is intended for coding only 512x512 grayscale images in RAW format (without heading). The used set of test images is accessible to downloading from the same address. The quality of decoded images was compared for CRs equal 8, 16, 32 and 64. As quality criterion, the peak signal to noise ratio was used:

$$PSNR = 10 \lg(255^2 / [\sum_{i=1}^I \sum_{j=1}^J (I_{ij} - I_{ij}^e)^2 / IJ])$$

Where I, J denote the image size, I_{ij} is the value of the ij -th pixel of original image, and I_{ij}^e defines the ij -th pixel value for the analyzed

(decompressed) image. Table 1 presents the obtained PSNRs for the considered methods.

Table 1. The quality of the test image compression for JPEG2000 and AGU, PSNR, dB

Image	CR=8		CR=16		CR=32		CR=64	
	JPEG2000	AGU	JPEG2000	AGU	JPEG2000	AGU	JPEG2000	AGU
Lenna	40.33	40.52	37.27	37.46	34.15	34.51	31.02	31.50
Barbara	38.07	39.26	32.87	34.65	28.89	30.77	25.87	27.55
Baboon	29.11	29.70	25.57	26.12	23.18	23.69	21.68	22.01
Goldhill	36.54	37.03	33.24	33.65	30.53	31.09	28.49	28.97
Peppers	38.17	38.33	35.80	35.55	33.54	33.32	30.79	30.90

As seen from data presented in Table 1, in overwhelming majority of the considered situations AGU outperforms JPEG2000 by quality of the decoded images. The only exceptions are CR=16 and CR=32 for the image Peppers for which JPEG2000 provides PSNRs that are better than for AGU by 0.2-0.25 dB. At the same time, for more complex images like Baboon and Goldhill the benefit of AGU for all CRs is 0.3-0.6 dB. And for the image Barbara that differs from other images by the presence of a large number of textural regions the advantage of AGU is 1.2-1.9 dB.

Image compression performance can be also compared for identical quality of decompressed images. For example, for PSNR=28.89 dB JPEG2000 compresses the image Barbara by 32 times while AGU compresses this image by 46.7 times, that is by 1.46 times better. For PSNR=25.87 dB AGU compresses this image by 101.5 times, that is 1.59 times better than JPEG2000 for which CR=64.

The presented data confirm that the proposed method outperforms JPEG2000 in image coding quality. The smoother is the image, the less difference of coding quality is observed for JPEG2000 and AGU. And the more complex and textural is the image, the difference of coding quality is larger. The fragment of decoded image Barbara for JPEG2000 and AGU is shown in Fig.5.



(a)



(b)

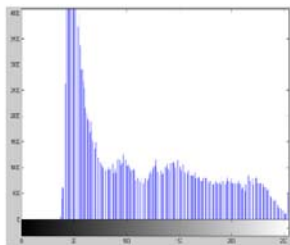
Fig. 5. A fragment of the decoded image Barbara, CR=32 a) JPEG2000, PSNR=28.89 dB b) AGU, PSNR=30.77 dB

VI. RESULTS

Image 1:
Original Image :



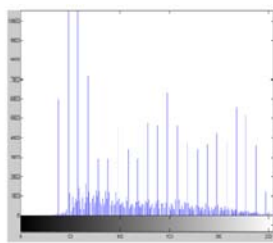
Histogram :



DCT compressed Image:



Histogram:



Error Image:



Histogram:

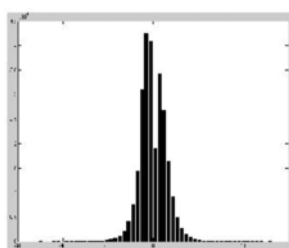
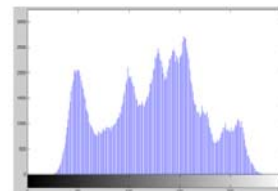


Image 2:
Original Image :



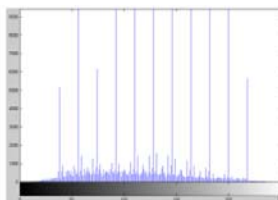
Histogram :



DCT compressed Image:



Histogram:



Error Image:



Histogram:

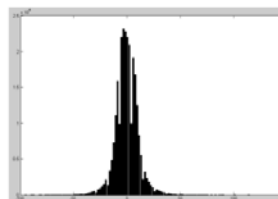


Table 2: Performance of DCT

Image	Compression Used	CR	MSE	PSNR
Image 1	DCT	51.1	108	27.78
Image 2	DCT	6.1	264.4	23.90

VII. CONCLUSIONS

The carried out studies show that the method proposed and described in this paper provides better quality of decoded images than JPEG2000 in most of practical situations. And its superiority for complex textured images in some cases can reach 1.9 dB.

The proposed method is obtained by rather simple modifications of JPEG, in which DCT serves as its core. This indicates that DCT is at least not worse transformation for use in image compression than DWT used as the basis of JPEG2000. For software realizations of AGU (not optimized), the required computation time is by about 15-20 times larger than for standard JPEG. The ways to speed up AGU can be studied in future. In particular, algorithms of fast integer

valued approximation of DCT in 32x32 blocks seem to lead to considerable decreasing of computation time.

In future it is possible to consider the use of partition schemes that make image compression methods more adaptive. Besides, a perspective direction is the use of DCT based image compression methods directed on reduction of blocking effect such as lapped orthogonal transforms

REFERENCES

- [1] Ahmed, N., Natarajan T., Rao K. R.: Discrete cosine transform. In: IEEE Transactions on Computers, Vol. 23, (1974) 90-93
- [2] Rao, K., Yip P.: Discrete Cosine Transform, Algorithms, Advantages, Applications. In: Academic Press (1990)
- [3] Wallace, G. K.: The JPEG Still Picture Compression Standard. In: Comm. Of the ACM, Vol.34, No.4 (1991)
- [4] Pennebaker, W. B., Mitchell, J. L.: JPEG Still Image Data Compression Standard. In: Van Nostrand Reinhold, New York (1993)
- [5] Christopoulos, C., Skodras, A., Ebrahimi, T.: The JPEG2000 still image coding system: an overview. In: IEEE Trans. on Consumer Electronics, Vol. 46, Issue: 4 (2000) 1103-1127
- [6] Taubman, D., Marcellin, M.: JPEG 2000: Image Compression Fundamentals, Standards and Practice. In: Boston: Kluwer (2002)
- [7] Egiazarian, K., Helsingius, M., Kuosmanen, P., Astola, J.: Removal of blocking and ringing artifacts using transform domain denoising. In: Proc. of ISCAS'99, Vol. 4, (1999) 139-142
- [8] Rissanen, J.: Generalized kraft inequality and arithmetic coding. In: IBM J. Res. Develop., Vol. 20, (1976) 198-203
- [9] Langdon, G.G., Rissanen, J.J.: A simple general binary source code. In: IEEE Trans. Inf.Theory, IT-28 (1982) 800-803
- [10] Yaroslavsky, L.: Local Adaptive Filtering in Transform Domain for Image Restoration, Enhancement and Target Location. In: 6th Int. Workshop on Digital Image Processing and Computer Graphics (DIP-97), SPIE volume 3346, (1997) 2-17
- [11] Ponomarenko N., Lukin V., Egiazarian K., Astola J.: Partition Schemes in DCT Based Image Compression. In: Technical Report 3-2002, ISBN 952-15-0811-6, Tampere University of Technology, Finland, (2002)
- [12] Malvar, H.S., Staelin, D.H.: The LOT: transform coding without blocking effects. In: IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 37, (1989) 553-559
- [13] Tran, T.D., de Queiroz, R., Nguyen, T.Q.: The generalized lapped biorthogonal transform. In: Proceedings of the ICASSP'98, Vol.3, (1998) 1441-1444