UDC 004.932
ORIGINAL SCIENTIFIC PAPER
RECEIVED: 27-05-2011
ACCEPTED: 14-07-2011

ACTA GRAPHICA 188

Regular Discrete Cosine Transform and its Application to Digital Images Representation

Author

Yuri A. Gadzhiev

Informational Systems Department,
Dagestan State Technical University (DSTU)
367030, Makhachkala, Russian Federation
e-mail: spritesoft@list.ru

Abstract:

Discrete cosine transform DCT-I, unlike DCT-II, does not concentrate the energy of a transformed vector sufficiently well, so it is not used practically for the purposes of digital image compression. By performing regular normalization of the basic cosine transform matrix, we obtain a discrete cosine transform which has the same cosine basis as DCT-I, coincides as DCT-I with its own inverse transform, but unlike DCT-I, it does not reduce the proper ability of cosine transform to the energy concentration. In this paper we consider briefly the properties of this transform, its possible integer implementation for the case of 8x8-matrix, its applications to the image itself and to the preliminary RGB colour space transformations, furthermore we investigate some models of quantization, perform an experiment for the estimation of the level of digital images compression and the quality achieved by use of this transform. This experiment shows that the transform can be sufficiently effective for practical use, but the question of its comparative effectiveness with respect to DCT-II remains open.

Keywords:

Image Compression, Frequency Transforms, DCT

1. Introduction

There exists a well known kind of linear matrix transforms called *discrete cosine transforms* (DCT) which is widely used in the field of digital signal processing and for lossy compression of halftone images in particular. The DCTs were introduced for the first time in *(Ahmed et al, 1974)*. There were introduced the DCTs of four forms depending on the cosine basis used. Second variety of these DCTs, known as DCT-II, with functional basis of the form $\cos(k(2n+1)\pi/2(N+1))$, *k,n=0,...,N*, is used most often. First variation, called DCT-I, which operates with common cosine basis $\cos(kn\pi/N)$, *k,n=0,...,N*, is not used commonly for the purposes noted above. This takes place in view of weak concentration of energy in the first component of transformed vector by the DCT-I versus the DCT-II (see discussion for some transforms energy concentration property in (*Salomon, 2001*). For instance, the vector of eight repeating values *(23,23,...,23)* results in the vector *(65.054,0,0,0,0,0,0,0)* by means of DCT-II and in the vector *(64.453,0, 5.092,0,5.092,0,5.092,0)* by means of DCT-I[1]. In the first case we can see that all components except the first one became zeroes, whereas in the second case yet 3 of the components are a nonzero. However, we can also observe that the approximations of a function given in points $f(0)=...=f(7)=23$ both

$$f(t)=\sum_{j=0}^{N} a_j \cos(jt\pi/N), \qquad (1)$$

and

$$f(t)=\sum_{j=0}^{N} a_j \cos(j(2t+1)\pi/2(N+1)), \qquad (2)$$

lead to one vector of the coefficients: *(a_0, a_1,..., a_7)=(23,0,0,0,0,0,0,0)*. This shows that the proper cosine transforms of both forms do not differ with respect to concentration of energy in the first component.

Now we have to consider the DCT-I in particular in order to know why its properties are so essentially different from those of cosine approximation of the form (1). The matrix of DCT-I is defined by the formula (see *Britanak et al., 2007)*, e.g.

$$C_{k,n}=\sqrt{\frac{1}{N}}\cdot\sqrt{\frac{2-\delta_{k,0}-\delta_{k,N}}{1+\delta_{n,0}+\delta_{n,N}}}\cdot\cos\left(\frac{kn\pi}{N}\right), \qquad (3)$$

k,n=0,...,N,

where $\delta_{i,j}=\{1 \text{ } when \text{ } i=j \text{ } and \text{ } 0 \text{ } otherwise\}$. This definition can be rewritten in the matrix form as

$$C=\sqrt{2N}^{-1}BSB,$$

where *S*= [$S_{k,n}$], $S_{k,n}$=cos(*knπ/N*), *k,n=0,...,N*, - cosine matrix, $B=diag(1,\sqrt{2},...,\sqrt{2},1)$ - diagonal (*N+1*)×(*N+1*) matrix. If we denote the outcome of DCT-I for some vector *x* as $c(x)$: $c(x)=Cx$ and, respectively, the outcome of cosine transform of *x* into the coefficients a_j, *j=0,...,N* by cosines in (1) as $s(x)$: $s(x)=S^{-1}x$, then, since $C=C^{-1}$ (the matrix *C* is orthogonal and symmetrical), we will have the following:

$$c(x)=Cx=C^{-1}x=\sqrt{2N}(BSB)^{-1}x=$$

$$=\sqrt{2N}B^{-1}S^{-1}B^{-1}x=\sqrt{2N}B^{-1}s(B^{-1}x).$$

Multiplication of $s(.)$ by $\sqrt{2N}B^{-1}$ in the right part of the last equation does not change the properties of cosine transform $s(.)$ since this action only scales the $c(.)$ outcome components. These properties are changing, as we can see in the last expression, owing to the multiplication of the original vector *x* by B^{-1} inside parentheses under $s(.)$, which clearly distorts the proportions of the vector *x* components *before* the application of the cosine transform $s(.)$ to it.

2. Basic cosine transform

Hereinafter we consider the transform $s(.)$ in details. Let *f* be a function given by its tabulated values $f_0, f_1, ..., f_N$ at the equidistant points *0,...,N*.

[1] Since the matrices of both transforms are orthogonal, their original and resulting vectors satisfy the equality $<x,x>=<y,y>$. Therefore we can check the result of calculation: $8\cdot 23^2 = 65.054^2 = 64.453^2 + 3\cdot 5.092^2$

$$f_i = c_0 \cos\left(0 \cdot \frac{i \cdot \pi}{N}\right) + c_1 \cos\left(1 \cdot \frac{i \cdot \pi}{N}\right) + \ldots + c_N \cos\left(N \cdot \frac{i \cdot \pi}{N}\right)$$

i.e. $$f_i = \Sigma_{m=0}^{N} c_m \cos\left(m \cdot \frac{i \cdot \pi}{N}\right).$$

For finding *k*-th coefficient c_k we multiply both parts of this equation by

$$\cos\left(k \cdot \frac{i \cdot \pi}{N}\right):$$

$$f_i \cos\left(k \cdot \frac{i \cdot \pi}{N}\right) =$$

$$= c_0 \cos\left(0 \cdot \frac{i \cdot \pi}{N}\right) \cos\left(k \cdot \frac{i \cdot \pi}{N}\right) + c_1 \cos\left(1 \cdot \frac{i \cdot \pi}{N}\right) \cos\left(k \cdot \frac{i \cdot \pi}{N}\right) + \ldots +$$

$$+ c_k \cos\left(k \cdot \frac{i \cdot \pi}{N}\right) \cos\left(k \cdot \frac{i \cdot \pi}{N}\right) + \ldots + c_N \cos\left(N \cdot \frac{i \cdot \pi}{N}\right) \cos\left(k \cdot \frac{i \cdot \pi}{N}\right)$$

or, in compact form,:

$$f_i \cos\left(k \cdot \frac{i \cdot \pi}{N}\right) = \Sigma_{m=0}^{N} c_m \cos\left(m \cdot \frac{i \cdot \pi}{N}\right) \cos\left(k \cdot \frac{i \cdot \pi}{N}\right).$$

Now we sum the values of every part this equality over $i=0,\ldots,(2N-1)$, i.e. over

$$\frac{i \cdot \pi}{N} = \frac{0 \cdot \pi}{N}, \frac{1 \cdot \pi}{N}, \ldots, \frac{(2N-1) \cdot \pi}{N}:$$

$$\Sigma_{i=0}^{2N-1} f_i \cos\left(k \cdot \frac{i \cdot \pi}{N}\right) = \Sigma_{i=0}^{2N-1} \Sigma_{m=0}^{N} c_m \cos\left(m \cdot \frac{i \cdot \pi}{N}\right) \cos\left(k \cdot \frac{i \cdot \pi}{N}\right) =$$

$$= \Sigma_{m=0}^{N} c_m \Sigma_{i=0}^{2N-1} \cos\left(m \cdot \frac{i \cdot \pi}{N}\right) \cos\left(k \cdot \frac{i \cdot \pi}{N}\right)$$

Let the sum $\Sigma_{i=0}^{2N-1} \cos\left(m \cdot \frac{i \cdot \pi}{N}\right) \cos\left(k \cdot \frac{i \cdot \pi}{N}\right)$ be denoted, for the sake of simplicity, by $\alpha_{m,k}$. It is well known that $\alpha_{0,0} = \alpha_{N,N} = 2N$, $\alpha_{1,1} = \alpha_{2,2} = \ldots = = \alpha_{N-1,N-1} = N$, and $\alpha_{m,k} = 0$ when $m \neq k$, (for $N=3$, e.g., the matrix A= $[\alpha_{m,k}]$ is

$$A = \begin{vmatrix} 6 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 6 \end{vmatrix}$$

Hence

$$\Sigma_{i=0}^{2N-1} f_i \cos\left(k \cdot \frac{i \cdot \pi}{N}\right) = \Sigma_{m=0}^{N} c_m \alpha_{m,k} = c_k \alpha_{k,k} ,$$

from whence

$$c_k = \frac{\Sigma_{i=0}^{2N-1} f_i \cos\left(k \cdot \frac{i \cdot \pi}{N}\right)}{\alpha_{k,k}}.$$

Since cos($k\pi$+x) = cos($k\pi$+x), then

$$\cos\left(k \cdot \frac{(N+t) \cdot \pi}{N}\right) = \cos\left(k \cdot \frac{(N-t) \cdot \pi}{N}\right).$$

By supposing $f_{N+t} = f_{N+t}$ (see draft at the Figure 1), what is necessarily in view of the cosine evenness, we get finally

$$c_k = \frac{f_0 + 2 \cdot \Sigma_{i=1}^{N-1} f_i \cos\left(k \cdot \frac{i \cdot \pi}{N}\right) + f_N \cdot \cos(k \cdot \pi)}{\alpha_{k,k}} =$$

$$= \frac{f_0 + 2 \cdot \Sigma_{i=1}^{N-1} f_i \cos\left(k \cdot \frac{i \cdot \pi}{N}\right) + f_N \cdot (-1)^k}{\alpha_{k,k}}$$

Let the coefficient by f_i in the expression for c_k be denoted by $r_{k,i}$. As we can see, these coefficients are defined by the expressions:

$$r_{0,0} = \frac{1}{2N}, r_{0,1} = r_{0,2} = \ldots = r_{0,N-1} = \frac{1}{N}, r_{0,N} = \frac{1}{2N}$$

...

$$r_{k,0} = \frac{1}{N}, \{r_{k,i} = \frac{2}{N} \cos\left(k \cdot \frac{i \cdot \pi}{N}\right), i = 1,\ldots,N-1\}, r_{k,N} = \frac{(-1)^k}{N} \qquad (4)$$

...

$$r_{N,0} = \frac{1}{2N}, \{r_{N,i} = \frac{(-1)^i}{N}, i = 1,\ldots,N-1\}, r_{N,N} = \frac{(-1)^N}{2N}$$

and the cosine transform *s*(.) matrix is $S^{-1} = [r_{k,n}]$.

It is easy to see that an analogue of Parseval equality for the cosine transform *s*(.) is the identity

$$f_0^{\,2} + 2\Sigma_{n=1}^{N-1} f_n^{\,2} + f_N^{\,2} = N(2c_0^{\,2} + \Sigma_{n=1}^{N-1} c_n^{\,2} + 2c_N^{\,2}).$$

Indeed, on the one hand,

$$\Sigma_{i=0}^{2N-1} f_i^{\,2} =$$

$$= \Sigma_{i=0}^{2N-1} [\Sigma_{n=0}^{N} c_n \cos\left(n \cdot \frac{i \cdot \pi}{N}\right) \cdot \Sigma_{m=0}^{N} c_m \cos\left(m \cdot \frac{i \cdot \pi}{N}\right)] =$$

$$= \Sigma_{i=0}^{2N-1} [\Sigma_{n=0}^{N} \Sigma_{m=0}^{N} c_n c_m \cos\left(n \cdot \frac{i \cdot \pi}{N}\right) \cos\left(m \cdot \frac{i \cdot \pi}{N}\right)] =$$

$$= \Sigma_{n=0}^{N}\Sigma_{m=0}^{N}c_n c_m \Sigma_{i=0}^{2N-1}\cos\left(n\cdot\frac{i\cdot\pi}{N}\right)\cos\left(m\cdot\frac{i\cdot\pi}{N}\right)=$$

$$= \Sigma_{n=0}^{N}\Sigma_{m=0}^{N}c_n c_m \alpha_{n,m} = N(2c_0{}^2 + \Sigma_{n=1}^{N-1}c_n{}^2 + 2c_N{}^2),$$

and, on the other hand,

$$\Sigma_{i=0}^{2N-1}f_i{}^2 = f_0{}^2 + 2\Sigma_{n=1}^{N-1}f_n{}^2 + f_N{}^2.$$

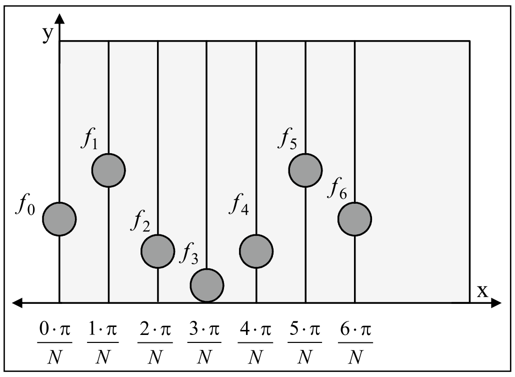

Figure 1. The graph illustrating relative location of the points of function for N=3.

3. Regular normalization of basic cosine transform $s(x)$

If we denote the cosine matrix as $S = [S_{k,i}]$,

$$S_{k,i} = \cos\left(k\cdot\frac{i\cdot\pi}{N}\right),$$

then the matrix S^{-1} of coefficients $r_{k,i}$ will be defined, as we can see directly from (4), by the matrix expression

$$S^{-1} = (2N)^{-1}B^2SB^2 = DSD,$$

where $D = \sqrt{2N}^{-1}B^2$. Multiplying both left and right parts of the last equation by S from left we get $I=SDSD$. This means $SD=(SD)^{-1}$. Now, let the transform $V=SD$ applied to x be denoted by $v(x)$. Then

$$v(x) = SDx = (SD)^{-1}x = D^{-1}S^{-1}x = D^{-1}s(x),$$

and we see from this that $v(x)$ unlike $c(x)$ does not distort the proportions of the components of x before an application of the basic cosine transform $s(.)$. So, the applying of $v(x)$ on the data vector mentioned above will result in (86.058,0,0,0,0,0,0,0)[2].

It is easy to see that for $v(x)$ an analogue of Parseval equation is the equation:

$$y_0{}^2 + 2\Sigma_{n=1}^{N-1}y_n{}^2 + y_N{}^2 = x_0{}^2 + 2\Sigma_{n=1}^{N-1}x_n{}^2 + x_N{}^2,$$

where x,y are original and transformed vectors. The matrix V of transform $v(x)$, as well as the matrix C of DCT-I, coincides with its own inverse matrix V^{-1} (the sets of its rows and columns are biorthogonal ones). However, unlike C it is both not symmetrical and not orthogonal. The elements of V are defined by the following formula (compare it with (3)):

$$V_{k,n} = \sqrt{\frac{1}{N}}\cdot\sqrt{\frac{2-\delta_{n,0}-\delta_{n,N}}{1+\delta_{n,0}+\delta_{n,N}}}\cdot\cos\left(\frac{kn\pi}{N}\right), \qquad (5)$$

$k,n=0,...,N$

Since $V=V^{-1}$, $V=T\Lambda T^{-1}= (T\Lambda T^{-1})^{-1} = T\Lambda^{-1}T^{-1}$, where Λ is diagonal matrix of eigenvalues of V. Hence, $\Lambda=\Lambda^{-1}$, what means, since Λ is diagonal, that absolute value of any of eigenvalues of V is equal to unit: $\lambda_i=\pm 1$, $i=0,...,N$. It is well known that the outcome of any linear transform V of normally distributed random vector x is distributed normally with a mean $V\mu$ and covariance matrix $V\Sigma V^T$, where $\mu=E(x)$ and $\Sigma=\text{cov}(x)$ are a vector of the means and covariance matrix of x, respectively. The transform V defined by (5) does not change covariance of x when its components are not correlated and $\text{cov}(x)=\Sigma_V=diag(2,1,...,1,2)$. In this case $V\Sigma_V V^T = \Sigma_V$.

4. Application of regular DCT to digital images representation

4.1. Regular DCT Implementation

For $N=7$ the matrix V in numbers is:

[2] $86.058 = \sqrt{7}\sqrt{2}\cdot 23$

$$V = \begin{bmatrix} 0.267 & 0.535 & 0.535 & 0.535 & 0.535 & 0.535 & 0.535 & 0.267 \\ 0.267 & 0.482 & 0.333 & 0.119 & -0.119 & -0.333 & -0.482 & -0.267 \\ 0.267 & 0.333 & -0.119 & -0.482 & -0.482 & -0.119 & 0.333 & 0.267 \\ 0.267 & 0.119 & -0.482 & -0.333 & 0.333 & 0.482 & -0.119 & -0.267 \\ 0.267 & -0.119 & -0.482 & 0.333 & 0.333 & -0.482 & -0.119 & 0.267 \\ 0.267 & -0.333 & -0.119 & 0.482 & -0.482 & 0.119 & 0.333 & -0.267 \\ 0.267 & -0.482 & 0.333 & -0.119 & -0.119 & 0.333 & -0.482 & 0.267 \\ 0.267 & -0.535 & 0.535 & -0.535 & 0.535 & -0.535 & 0.535 & -0.267 \end{bmatrix}$$

Following C++ code illustrates a simplest integer algorithm for $v(x)$ with it:

```
#define SCALE (1<<12)
//const LONG
c0=roundoff((sqrt(2.0)/
sqrt(N))*SCALE);
#define c0 2189

//const LONG c1=roundoff((1.0/
sqrt(2*N))*SCALE);
#define c1 1095

//const LONG
c2=roundoff((sqrt(2.0)*cos(M_PI/N)/
sqrt(N))*SCALE);
#define c2 1973

//const LONG c3=roundoff((sqrt(2.0)
*cos(3.0*M_PI/N)/sqrt(N))*SCALE);
#define c3 487

//const LONG c4=roundoff((sqrt(2.0)
*cos(2.0*M_PI/N)/sqrt(N))*SCALE);
#define c4 1365

inline void DCT(LONG &x0, LONG &x1,
LONG &x2, LONG &x3, LONG &x4, LONG
&x5, LONG &x6, LONG &x7)
{

   register LONG c1p07=c1*(x0+x7);
   register LONG c1m07=c1*(x0-x7);

   register LONG p16=x1+x6;
   register LONG m16=x1-x6;

   register LONG p25=x2+x5;
   register LONG m25=x2-x5;

   register LONG p34=x3+x4;
   register LONG m34=x3-x4;

   x0=c1p07+c0*(p16+p25+p34);
   x1=c1m07+c2*m16+c4*m25+c3*m34;
   x2=c1p07+c4*p16-c3*p25-c2*p34;
   x3=c1m07+c3*m16-c2*m25-c4*m34;
   x4=c1p07-c3*p16-c2*p25+c4*p34;
   x5=c1m07-c4*m16-c3*m25+c2*m34;
   x6=c1p07-c2*p16+c4*p25-c3*p34;
   x7=c1m07+c0*(-m16+m25-m34);
}
```

This algorithm used for DCT transforms some colours of 8-bytes sequence of pixels'. After each run of DCT its outcome has to be renormalized by means of dividing by SCALE (in this case it is equal to $2^{12} = 4096$) and then rounded off.

General algorithm used for the representation of an image in compressed form is the following:

1. Every pixel of image is subjected to RGB colour space transform.

2. The entire image is separated on the blocks of 8x8 pixels and each of these blocks is successively subjected to regular DCT for each of 3 components new colour space.

3. All transformed pixel values obtained in the previous step are quantized and saved into the outer file in the compressed form (all (0,0) – block elements are subjected to different transform since their absolute values are usually significantly greater than others).

In order to end the compression on the latter step any compression algorithm can be used. However order-0 locally adaptive schemes are more preferable. We used the so-called parametrically adaptive coding (PAC) described in detail in *(Gadzhiev, 2001)*. The PAC-coder is simple order-0 coder, which uses move-to-front transform (MTF), outcome of which is then encoded by the PAC-algorithm.

Further on, we will describe in detail our approaches to RGB colour space transform and quantization.

3.2. RGB COLOUR SPACE TRANSFORM

For preliminary RGB colour space transform we propose the use of the same regular DCT with *N=2*, normalized by means of dividing by $\sqrt{2}\sqrt{N} = 2$. New colour space consists of three components, first of which is an intensity component and two others are mixed colour components (we denote their by Y,C1,C2, respectively). The matrix V'_3 of the transform of RGB vector into YC1C2 is

$$V'_3 = \begin{bmatrix} 0.25 & 0.5 & 0.25 \\ 0.25 & 0 & -0.25 \\ 0.25 & -0.5 & 0.25 \end{bmatrix}$$

and the matrix of inverse transform (YC1C2 into RGB) is

$$\left(V'_3\right)^{-1} = \begin{bmatrix} 1 & 2 & 1 \\ 1 & 0 & 1 \\ 1 & 2 & 1 \end{bmatrix}$$

The latter transform can be realized very effectively and that is its biggest advantage.

3.3. QUANTIZATION MODELLING

The simplest expression of the form Quantization of the outcome of $v(x)$ can be done by a simple expression of the form

$$c(i+d)(j+d) \qquad (6)$$

where *i, j* are coordinates of pixel inside of a block, *c* and *d* are constants which control both the achieved compression level and the image quality. Reasonable quality for illustrative images inside a text can be achieved with $c \approx 1$, $1 \leq d \leq 3$ for Y and $3 \leq d \leq 9$ for Cb and Cr components of YCbCr colour space. However, we can get significantly better results with use of the following approach for the quantization modelling.

It is obvious that correct quantization does not break any of the properties of used transform, and the property to concentrate the energy of data in first component of transformed vector in particular. Therefore, it is allowable for the quantization of each *i*-th component of the DCT outcome by the coefficient q_i to use instead of V the matrix of the form $diag(q)V$, where q is some number vector. For two-dimensional transform we use two stage processing. On the first stage set of the block's row is transformed, and on the second stage the columns of coefficients obtained by the first step transform are also subjected to such a transform. This process is equivalent to the transform VDV^T for square data block D. Suppose that the first stage quantization vector is r and on the second stage it is q. Then full processing is described by the expression $diag(q)V(diag(r)VD^T)^T$ that is equivalent to the expression $diag(q)VDV^Tdiag(r)$. If we denote elementwise product of some matrices A, B by $A \times B$ ($(A \times B)_{i,j} = A_{i,j} B_{i,j}$) then $diag(q)\, VDV^Tdiag(r) = VDV^T \times Q$, where $Q = qr^T$ square quantization matrix. (Indeed, for any matrix product AB we have $AB \times Q = (diag(q) \cdot A)\,(diag(r) \cdot B)$ since if $AB \underset{den}{=} C$ then $C_{i,j} = \Sigma_k A_{i,k} B_{k,j}$ and, respectively, $C_{i,j}Q_{i,j} = C_{i,j}q_i r_j = q_i r_j \Sigma_k A_{i,k} B_{k,j} = \Sigma_k (q_i A_{i,k})(r_j B_{k,j})$). That is any correct quantization table Q can be expressed as qr^T, where r and q are the quantization vectors for the first and second stages of the transform process, respectively. In the case of (6) $q=r$, where $q_i^{-1} = r_i^{-1} = \sqrt{c}(i+d)$. Generally, having denoted for the sake of simplicity the vector of reciprocals by the prime symbol, $u' = (u_i^{-1})$, we can model any monotone quantization's dependence by the expression of the form $u'_i = d(i+1)^p + k$. Respectively, $r'_i = d_2(i+1)^{p_2} + k_2$ and $q'_i = d_1(i+1)^{p_1} + k_1$ In the terms of $u'_{\min} = \min_i(u'_i) = u'_0$, $u'_{\max} = \max_i(u'_i) = u'_N$ we have $u'_0 = d + k$ and

$$u'_N = d(N+1)^p + k = d(N+1)^p + u'_0 - d = \\ = d((N+1)^p - 1) + u'_0.$$

Hence $d_1 = \dfrac{q'_N - q'_0}{(N+1)^{p_1} - 1}$, $d_2 = \dfrac{r'_N - r'_0}{(N+1)^{p_2} - 1}$

and $k_1 = q'_0 - d_1$, $k_2 = r'_0 - d_2$.

For example, if we take $p_1 := 1$, $q'_0 := \sqrt{16}$, $q'_N := 5$, $p_2 := 3$, $r'_0 := \sqrt{15}$, $r'_N := 70$ then,

$d_1 = 0.143$, $k_1 = 3.857$,

$$q'^T = [\,4 \quad 4.143 \quad 4.286 \quad 4.429 \quad 4.571 \quad 4.714 \quad 4.857 \quad 5\,]$$

$d_2 = 0.129$, $k_2 = 3.744$,

$$r'^T = [\,3.873 \quad 4.779 \quad 7.238 \quad 12.026 \quad 19.919 \quad 31.696 \quad 48.13 \quad 70\,]$$

and rounded off quantization matrix is

$$q' \cdot r'^T = \begin{bmatrix} 15 & 19 & 29 & 48 & 80 & 127 & 193 & 280 \\ 16 & 20 & 30 & 50 & 83 & 131 & 199 & 290 \\ 17 & 20 & 31 & 52 & 85 & 136 & 206 & 300 \\ 17 & 21 & 32 & 53 & 88 & 140 & 213 & 310 \\ 18 & 22 & 33 & 55 & 91 & 145 & 220 & 320 \\ 18 & 23 & 34 & 57 & 94 & 149 & 227 & 330 \\ 19 & 23 & 35 & 58 & 97 & 154 & 234 & 340 \\ 19 & 24 & 36 & 60 & 100 & 158 & 241 & 350 \end{bmatrix} \qquad (7)$$

Diagonal elements alteration trend of this quantization matrix illustrated on *figure 2*.

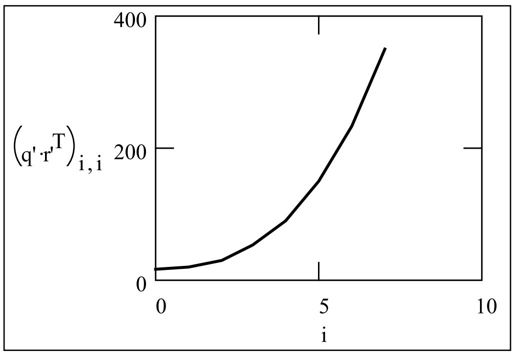

Figure 2. The elements alteration trend of quantization matrix (7)

We use this matrix for quantization of Y-component of YC1C2 color space. For C1 and C2 components we use just the same approach with the parameters $p_1 := 1$, $q'_0 := \sqrt{25}$, $q'_N := 10$ $p_2 := -0.5$ $r'_0 := \sqrt{25}$, $r'_N := 70$, for C1 and $p_1 := 1$, $q'_0 := \sqrt{49}$, $q'_N := 15$, $p_2 := -0.5$, $r'_0 := \sqrt{49}$, $r'_N := 25$ for C2.

For C1 quantization we get results as follows:

$d_1 = 0.714$, $k_1 = 4.286$,

$$q'^T = [\,5 \quad 5.714 \quad 6.429 \quad 7.143 \quad 7.857 \quad 8.571 \quad 9.286 \quad 10\,]$$

$d_2 = -100.55$, $k_2 = 105.55$,

$$r'^T = [\,5 \quad 34.45 \quad 47.497 \quad 55.275 \quad 60.582 \quad 64.5 \quad 67.54 \quad 70$$

$$q' \cdot r'^T = \begin{bmatrix} 25 & 172 & 237 & 276 & 303 & 323 & 338 & 350 \\ 29 & 197 & 271 & 316 & 346 & 369 & 386 & 400 \\ 32 & 221 & 305 & 355 & 389 & 415 & 434 & 450 \\ 36 & 246 & 339 & 395 & 433 & 461 & 482 & 500 \\ 39 & 271 & 373 & 434 & 476 & 507 & 531 & 550 \\ 43 & 295 & 407 & 474 & 519 & 553 & 579 & 600 \\ 46 & 320 & 441 & 513 & 563 & 599 & 627 & 650 \\ 50 & 345 & 475 & 553 & 606 & 645 & 675 & 700 \end{bmatrix} \qquad (8)$$

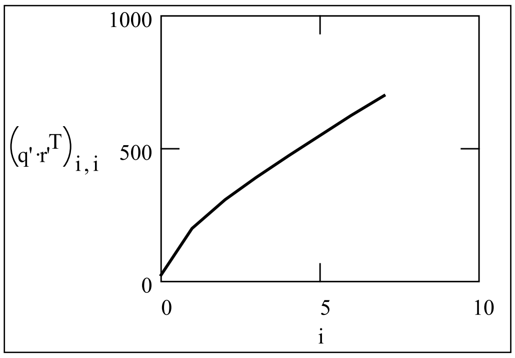

Figure 3. elements alteration trend of quantization matrix (8) for C1

For C2 quantization we get results as follows:

$d_1 = 1.143$, $k_1 = 5.857$,

$$q'^T = [\,7 \quad 8.143 \quad 9.286 \quad 10.429 \quad 11.571 \quad 12.714 \quad 13.857 \quad 15\,]$$

$d_2 = -27.845$, $k_2 = 34.845$,

$$r'^T = [\,7 \quad 15.155 \quad 18.768 \quad 20.922 \quad 22.392 \quad 23.477 \quad 24.32 \quad 25\,]$$

$$q' \cdot r'^T = \begin{bmatrix} 49 & 106 & 131 & 146 & 157 & 164 & 170 & 175 \\ 57 & 123 & 153 & 170 & 182 & 191 & 198 & 204 \\ 65 & 141 & 174 & 194 & 208 & 218 & 226 & 232 \\ 73 & 158 & 196 & 218 & 234 & 245 & 254 & 261 \\ 81 & 175 & 217 & 242 & 259 & 272 & 281 & 289 \\ 89 & 193 & 239 & 266 & 285 & 298 & 309 & 318 \\ 97 & 210 & 260 & 290 & 310 & 325 & 337 & 346 \\ 105 & 227 & 282 & 314 & 336 & 352 & 365 & 375 \end{bmatrix} \qquad (9)$$

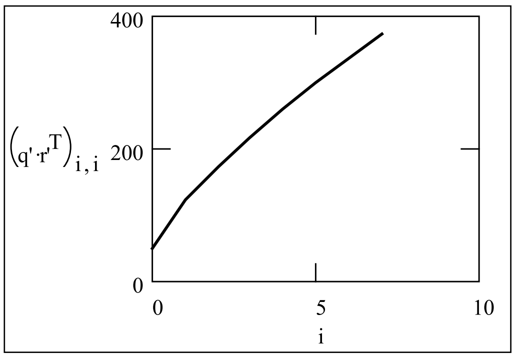

Figure 4. elements alteration trend of quantization matrix (8) for C2

3.4. Experimental results

For the experiments we used Miscellaneous Test Images database obtained from the website of SIPI (Signal and Image Processing Institute, University of Southern California) by the reference http://sipi.usc.edu/database/misc.zip. This zip-file (size 12.4 MB) consists of 44 images, 16 colour and 28 monochrome.

We show the results of the RGB colour space transform described above and the quantization approach to one of the test images for which the quantization distortions are usually clearly visible.

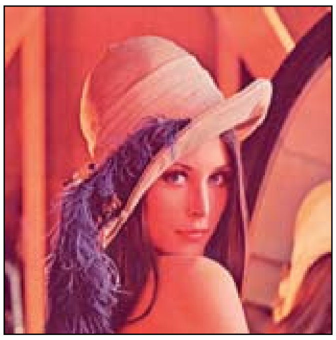

Figure 5. Regular DCT quantized with QFactor=0.5 image (PSNR= 34.52 dB, H0=0.504503 b/s, PACLength= 37 192 byte, Ratio=0.0473)

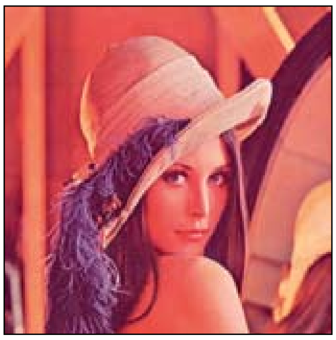

Figure 6. Regular DCT quantized with QFactor=1.0 image (PSNR= 32.69 dB, H0= 0.315169 b/s, PACLength = 22 362 byte, Ratio=0.0284)

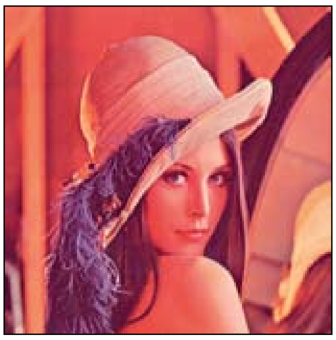

Figure 9. Original image file 4.2.04. bmp (H0=7.75066 b/s, Length = 786 486 byte)

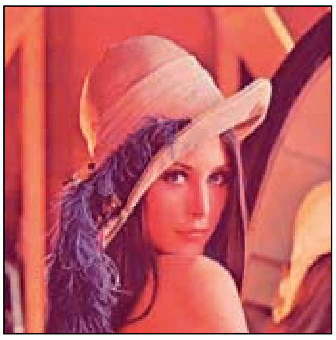

Figure 7. Regular DCT quantized with QFactor=2.0 image (PSNR= 30.85 dB, H0= 0.193717 b/s, PACLength = 13 096 byte, Ratio=0.0167)

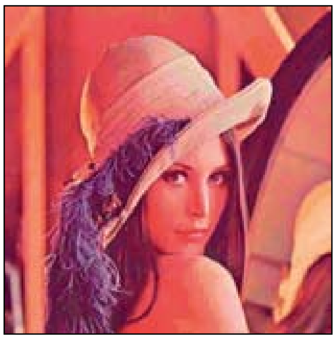

Figure 8. Regular DCT quantized with QFactor=3.0 image (PSNR= 29.80 dB, H0= 0.148427 b/s, PACLength = 9 910 byte, Ratio=0.0126)

The QFactor is here the quantization factor, i.e. the factor of matrices (7),(8),(9), which allows to control the compression level (and the image quality respectively); PSNR is the Peak Signal-to-Noise Ratio for the Y-component of YCbCr colour space calculated by the formula, where is the number of pixels in the compared files, and - the values of Y-component for the couple of pixels with coordinates from these two files; is order-0 binary entropy of the file before compressing (if the file is compressed) expressed in the bits per symbol units (b/s) which is calculated by the formula , where is the number of bytes in the file that are equal to and is the length of the file in bytes; "PACLength" is the length of compressed PAC-file and "Ratio" is the compression ratio calculated as the ratio between compressed and original file lengths in bytes.

Within the framework of this experiment we have also done a comparison between regular DCT described here on one hand and the JPEG image compression on the other. For this purpose we saved each image from test archive as 24-bits colour BMP file with help of standard Windows utility MSPAINT.EXE (version 5.1, build 2600.xpsp1). Then we had been opening successively each of these BMP files in the MSPAINT and then saving it as JPEG. Besides this we transformed these BMP files by means of described in this paper algorithm with quantization factor QFactor=1.0 into compressed PAC files. In the table 1 below we show the results of this experiment.

All PAC images from together with PAC-coder and the programs for execution used are available on-line at http://spritesoft.narod.ru/dctrtest/misc.zip. (1.23 Mb).

As a conclusion to this part of the presentation we will show one more image from another test image database (resided at the WEB-site http://testimages.tecnick.com), for the purpose to demonstrate what kind of visual image quality is obtained by means of approaches described above when a good-quality original photo is used (see Fig. 10 below).

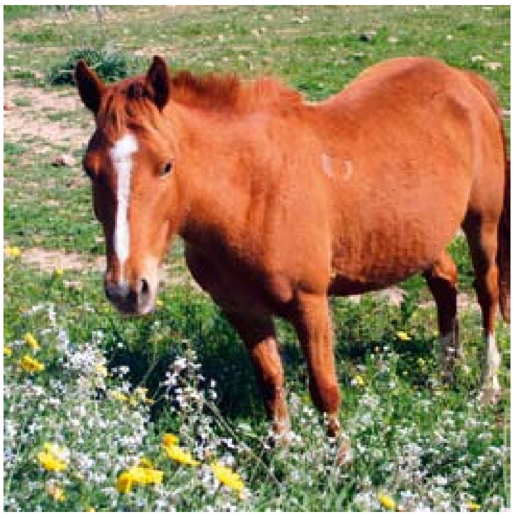

Figure 7. Test image RGB_OR_1200x1200_062.bmp (BMPLength= 4 320 054 byte, H0=7.90837), regular DCT representation with QFactor=1 (PSNRY= 29.22 dB, H0= 0.595859, PAC-Length=227 496 bytes, Compression Ratio=0.0527).

Table 1. Comparison of the average values for image database compresed with Reqular DCT and JPEG .

BMP 24-bit original file size (bytes)	Regular DCT PAC compressed			JPEG compressed		
	PAC file size (bytes)	Compression ratio	PSNR for the Y-component of YcbCr color space (dB)	JPEG file size (bytes)	Compression ratio	PSNR for the Y-component of YcbCr color space (dB)
1155126	35966	0,0343	31,86	60365	0,05635	36,20

4. Conclusion

In this paper we introduced a discrete form of cosine transform which uses the simplest cosine basis, however, unlike DCT-I, it remains applicable for compression. The matrix of this transform has less complicated numerical structure than the DCT-II, and at the same time, reasonable compression level and representation quality are obtained when simple regularized expressions for the quantization are used. For lossy compression of digital images this transform is certainly more preferable than DCT-I, but the question of its effectiveness in comparison with DCT-II is open.

References

AHMED, N., NATARAJAN, T., RAO, K. R., 1974, "Discrete Cosine Transform", IEEE Trans. Computers, C-23 , pp. 90-93.

BRITANAK V., YIP P., RAO K.R., 2007, "Discrete cosine and sine transforms", Academic Press, 349 pp.

EULER, L., 1961, "Vvedenie v analiz beskonechnyh" ("Introduction to analysis of infinities"), Vol. I., M., GIFML, 315 pp.

GADZHIEV, YU. A., 2001, "Posledovatelnoe adaptivnoe kodirovanie v parametricheski opredelennoi sisteme dvoichyh kodov dlja primenenija v algoritmah LZ-kompressii" ("Successive adaptive coding by means of a parametrically defined system of binary codes for use with LZ-compression algorithms"), Makhachkala, 192 pp., (Thesis).

SALOMON, D. A, 2001, "Guide to Data Compression Methods", Springer-Verlag, 295 pp.