VOL. 10, NO. 8, MAY 2015 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

A NOVEL METHOD FOR REDUCING NUMBER OF COMPUTATION IN 2D-DCT

K. K. Senthilkumar[1], R. Seshasayanan[1] and D. Gayathri[2]
[1]Department of Electronics and Communication Engineering, CEG, Anna University, Chennai, Tamilnadu, India
[2]Prince Sri Venkateshwara Padmavathy Engineering College, Chennai, Tamilnadu, India
E-Mail: senkrilak@gmail.com

ABSTRACT

The DCT performs very important role in the application of lossy compression for representing the pixel values of an image using lesser number of coefficients. Recently, many algorithms have been devised to compute DCT. In the initial stage of image compression, the image is generally subdivided into smaller sub-blocks and then these sub-blocks are applied to DCT. In this paper we have presented a novel method for DCT computation to reduce the number of computations based on the difference between the pixel values of adjacent rows. The DCT computations for second row are replaced by first row DCT computations when all the pixel values of second row are having very less difference from the first row pixels. In this way a larger number of computations are reduced. The method is verified with various high and less correlated images and the result shows that image quality is not much affected even though 4 bits per pixel are considered for row comparison. The correlation between the pixels of two rows is calculated by fixing a threshold value which depends on the elimination of number of bits used for row pixel comparison. The simulation results shows that the proposed DCT method reduces the Number of computations by 50.02 % for highly correlated images and reduces 23.63 % for less correlated images without much affecting the image quality.

Keywords: image compression, DCT, IDCT, CIM.

1. INTRODUCTION

Image compression is a process of reducing the size of representation in binary format for graphics file without affecting the quality of the image to an objectionable level and this reduction helps to store more images for the same amount of storage device. It also decreases the transmission time for images to be sent over the various technologies like Internet [1, 5, 3, 8]. The Discrete Cosine Transformation, which is the most widely used technique today for image compression, was initially defined in [1]. It came up as a revolutionary standard when compared with the other existing transforms. After that an algorithm for computing Fast Discrete Cosine Transformation (FDCT) was introduced by Chen *et al*., in [5] which were based on matrix decomposition of the orthogonal basis function of the cosine transform. The method took (3N/2) ($\log_2 N$ - 1) +2 real additions and $N \log_2 N$ - 3N/2 + 4 real multiplications, this is approximately six times faster than the conventional approach. As a next step a new algorithm was introduced for the 2m-point discrete cosine transform as in [3]. This algorithm reduced the number of multiplications to half of those required by the existing efficient algorithms (12 multiplications and 29 additions), and it makes the system simpler by decomposing the N-point IDCT into the sum of two N/2-point IDCTs. In the same period another algorithm was also given as in [20], which was based on the technique of divide and conquer. But it also kept the number of multiplications and additions as 12 and 29 respectively. A recursive algorithm for DCT [9] was presented with a structure that allows the generation of the next higher order DCT from two identical lower order DCT's to reduce the number of adders and multipliers (12 multiplications and 29 additions). Loffler came up with a practical fast 1-D DCT algorithm [11] in which the number of multiplications was reduced to 11 by inverting add/subtract modules and finds an equivalence for the rotation block (only 3 additions and 3 multiplications per block instead of 4 multiplications and 2 additions). Later, many algorithms were constantly introduced to optimize the DCT implementation. In recent years, the idea of implementing DCT using CORDIC [13] (CO-ordinate Rotation Digital Computer) which uses only shift and add arithmetic with look-up-tables. Another technique called Distributed Arithmetic was devised [14] which computes multiplication as distributed over bit-level memories and adders. A method combining fractal image compression along with DCT is proposed in [17]. ROM free 1D DCT architecture discussed in [15] and this architecture is based on distributive arithmetic (DA) method and compares the area and power reduction.

Algorithm to implement Distributed arithmetic using MUX and generic gates to reduce sign extension error is given in [14]. Many algorithms were proposed based on modifications done in the Loeffler algorithm. As in [6,18], unsigned constant coefficient multiplication is done by moving two negative signs to the next adder to make them positive and it was implemented using multiplier less operation. And in [2] to reduce the resources usage and to increase the maximum frequency by rearranging the ADD blocks to the consecutive stages. The prime N-length DCT was divided into similar cyclic convolution structures and the DCT was implemented using systolic array structure [4]. Also to eliminate the use of multipliers by using shift and addition operations, many algorithms were devised.

A technique uses Ramanujan numbers for calculating cosine values and uses Chebyshev type recursion to compute the DCT [7]. A recursive algorithm to compute the DCT with less complexity was introduced

with the use of a recursive kernel [13]. Using Algebraic Signal Approach (with polynomial algebras) factorization is applied recursively to obtain another fast algorithm as mentioned in [19]. Using some definite properties of Cordic algorithm, a high quality low power Cordic based Loeffler DCT architecture is proposed in [18].Other techniques for image compression include the use of fractals and wavelets. These methods have not gained widespread acceptance for use on the Internet's of this writing. However, both methods offer promise because they offer higher compression ratio than the JPEG or GIF methods for certain types of images. Another new method that may in time replace the GIF format is the PNG format. Compressing an image is significantly different than compressing raw binary data. Of course, general purpose compression programs can be used to compress images, but the result is less than optimal. This is because images have certain statistical properties which can be exploited by encoders specifically designed for them. Also, some of the finer details in the image can be sacrificed for the sake of saving a little more bandwidth or storage space. This also means that lossy compression techniques can be used in this area [12].

A multilevel wavelet transform is proposed in [16] for video compression instead of block level DCT and there is no significance for reducing the computations. Normally in DCT process, the computation can be computed either 1D or 2D directly for 8 x 8 images. The 64 pixels of any 8x8 image have mostly same values if it is a background image. If the values have either same or small allowable difference in 1st row and second row pixels, it is unnecessary to compute the DCT for second row and use the values from 1st row computation.

This paper proposes a new method that computes the DCT based on the difference between pixels of two rows and also it reduces the computations. The paper is organized as follows: The review of DCT algorithms works is given in section 2, DCT using the proposed comparative input method is discussed in section 3, the performance analysis and comparative analysis of the proposed DCT computation method is given in section 4 and finally, the conclusion is discussed in section 5.

2. ALGORITHMS FOR DCT IMPLEMENTATION

There are generally 2 methods for computing the 2-D DCT:

(i) Direct 2-D computation
(ii) Decomposition into two 1-D DCTs.

We are adapting the second approach to compute the 2-D DCT. The row transformation is initially applied to obtain a 1-D output and then applying it the next time yields the 2-D output as shown in Figure-1.

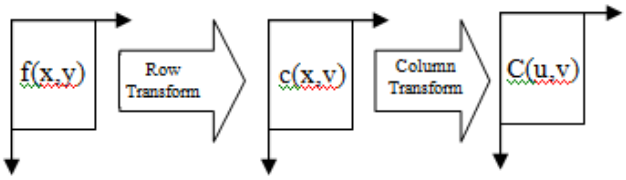

Figure-1. Decomposition of 2-D DCT.

Initially, the DCT equations are considered before transforming in the form of algorithm.

The 2-Dimensional DCT equations are given by

$$F(u,v) = D(\mathrm{u})D(\mathrm{v})\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\cos\left[\frac{(2x+1)u\pi}{2N}\right]\cos\left[\frac{(2y+1)v\pi}{2N}\right] \quad (1)$$

Where u, v= 0,1,2,3,……,N-1

$D(\mathrm{v}) = D(\mathrm{u}) = \sqrt{1/N}$ for u,v = 0
$D(\mathrm{v}) = D(\mathrm{u}) = \sqrt{2/N}$ for u,v = 1,2,3……,(N-1)

In this equation(1), for a sub block size NxN, *f(x,y)* is the input matrix. *F(u,v)* is the 2-D DCT output. *D*(u) and *D*(v) are the normalizing factors. Both the cosine terms represent the orthonormal basis functions which are nothing but the standard representation of the cosine functions which are used to map the input pixels into the transformed coefficiens. The input values need to be just multiplied with the orthonormal basis functions and the normalizing factor to get the DCT output. The 1-dimensional DCT equations are given by the equation (2)

$$F(u) = D(\mathrm{u})\sum_{x=0}^{N-1} f(x)\cos\left[\frac{(2x+1)u\pi}{2N}\right] \quad (2)$$

for u=0,1,2…N-1
$D(\mathrm{u}) = \sqrt{1/N}$ for u = 0
$D(\mathrm{u}) = \sqrt{2/N}$ for u = 1,2,3……., (N-1)

Here f(x) is the 1-D row input. The cosine term is the orthonormal function. F(u) gives the 1-D DCT output. D (u) is the normalizing factor.

To implement the DCT, we are using the modified Lee algorithm [3]. In this algorithm, The DCT computation is decomposed into 3 steps and mathematical simplifications are applied. This results in the 1-D DCT output. Also the implementation is done using the basic Chen's [5] algorithm and comparison is done to show that the modified Lee algorithm reduces the complexity by 46%.

A. Fast algorithm

The algorithm proposed by Chen.et.al. [5] called 'Fast algorithm'. This computes the 2-D DCT in terms of two 1-D DCTs. The computation is done as said earlier computing 1-D DCT, transposing, computing 2-D DCT. For 2-D DCT, the 8x8 transformation matrix corresponding to the 8x8 basis function is given by

$$A=\begin{bmatrix} d & d & d & d & d & d & d & d \\ a & c & e & g & -g & -e & -c & -a \\ b & f & -f & -b & -b & -f & f & b \\ c & -g & -a & -e & e & a & g & -c \\ d & -d & -d & d & d & -d & -d & d \\ e & -a & g & c & -c & -g & a & -e \\ f & -b & b & -f & -f & b & -b & f \\ g & -e & c & -a & a & -c & e & -g \end{bmatrix}$$

Where b = C_1, c=C_2, d=C_3, a= C_4, e=C_5, f=C_6, g=C_7;

$$C_i = 0.5\cos(i\pi/16)$$

But in Chen's algorithm, the 8x8 transformation matrix is decomposed into two 4x4 matrices. This is done by considering the input values which need to be multiplied with common coefficients (in the transformation matrix). After decomposing the two 4x4 transformation matrices obtained are,

$$\begin{pmatrix} Y(0,:) \\ Y(2,:) \\ Y(4,:) \\ Y(6,:) \end{pmatrix} = \begin{pmatrix} a & a & a & a \\ c & f & -f & -c \\ a & -a & -a & a \\ f & -c & c & -f \end{pmatrix} \begin{pmatrix} X(0,:)+X(7:,) \\ X(1,:)+X(6:,) \\ X(2,:)+X(5:,) \\ X(3,:)+X(4:,) \end{pmatrix}$$

$$\begin{pmatrix} Y(1,:) \\ Y(3,:) \\ Y(5,:) \\ Y(7,:) \end{pmatrix} = \begin{pmatrix} b & d & e & g \\ d & -g & -b & -e \\ e & -b & g & d \\ g & -e & d & -b \end{pmatrix} \begin{pmatrix} X(0,:)-X(7:,) \\ X(1,:)-X(6:,) \\ X(2,:)-X(5:,) \\ X(3,:)-X(4:,) \end{pmatrix}$$

The X corresponds to the 1-D input values and Y corresponds to the 1-D output values. The equations of Chen are implemented using the signal flow graph shown in Figure-2. The number of computations involved are (3N/2) (log2 N - 1) +2 real additions and (N log2 (N)) - 3N/2 -t 4 real multiplications. Hence for N=8, it requires 16 multiplications and 12 additions.

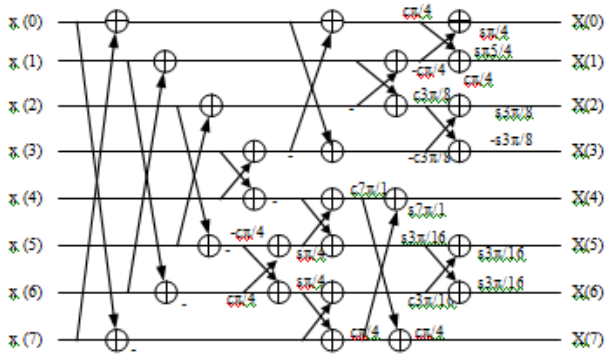

Figure-2. Signal flow graph of Fast algorithm.

3. IMAGE COMPRESSION USING PROPOSED DCT

The overall image compression process for any image is carried out by performing the steps as shown in Figure-3. Initially, an input image is considered for compressing and it has to be subdivided into smaller sub blocks.

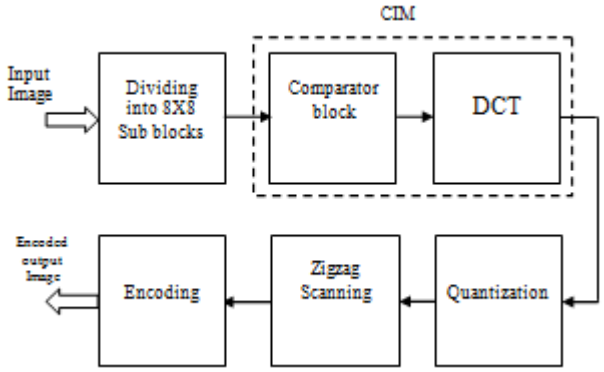

Figure-3. Image compression process.

In most applications, images are subdivided because correlation (redundancy) between adjacent sub images is reduced to some acceptable level and so that n is an integer power of 2, whereas before, n is the sub image dimension. The latter condition simplifies the computation of the sub image transforms. In general, both the level of computational complexity and compression increases as the sub image size increases.

The most popular sub image sizes are 8x8 and 16x16. In our process, we consider the sub-division of images into 8x8 sizes to ease the process. Also the frequency transformations like DCT is good at compressing rather smooth areas with low frequency content, but quite bad at compressing high frequency content areas. Any matrices of sizes greater than 8X8 are harder to do mathematical operations or not supported by hardware or take longer time. Those are designed so that they can be implemented using parallel hardware. Each block is independent, and can be calculated on a different computing node, or shared out to as many nodes. So, parallel computing on that level was very unusual for JPEG standard. Any matrices less than 8X8 have enough into to continue along with the pipeline. And it will consume more coefficients, so we go for dividing 8X8 sub-blocks method. It uses lesser number of bits as compared to the original representation.

The subdivided blocks are generally applied directly to the next corresponding steps, that is, the transformation, quantization and the encoding. But in our paper, we present a different approach after performing the sub division. We introduce a new method based on comparing the input values itself. After performing this Comparative Input Methods (CIM), the other steps of the compression process are carried out.

A. DCT Process through Comparative Input Method (CIM)

This step revolves around a new approach of comparing the input pixels before applying to the next step, which is the one dimensional DCT process. In this step, each 8x8 block of the input image, which is obtained through sub division process.

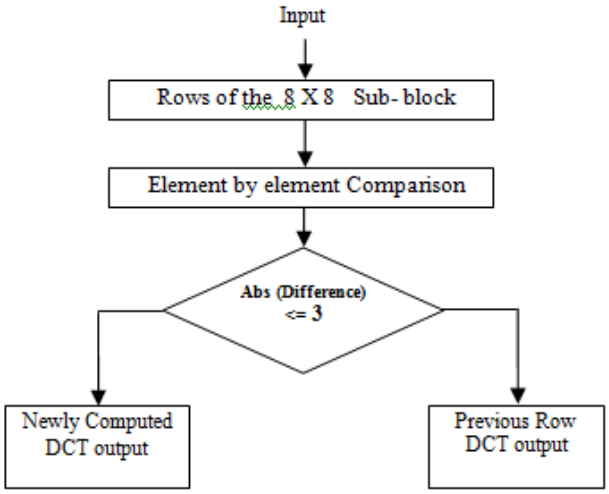

Figure-4. Flow chart for Comparative Input Method (CIM).

In general, each and every row (an array of 8 elements) is applied as input to the 1-D DCT to obtain an output array of 8 elements. But this step is modified in our method. Considering an 8x8 block, row separation is applied as the initial step. As a result, each of the 8 rows is obtained separately. Then one dimensional DCT is applied for the 1st row and the corresponding output array is obtained. And then, from the 2nd row onwards, each row is compared with the previous row. This comparison is done for all the eight element of the rows. If each and every element of a row is found to be nearly same as the one in the previous row, the next step need not be performed for that particular 2nd row. The previous row's DCT output is considered as output instead of 2nd row's DCT output. If the comparison fails and the elements are not matched, then the 1-D DCT has to be applied again for that particular row to obtain a new output array. This is done for all the remaining seven rows of the 8x8 sub block. The flow of this comparison process is explained in Figure-4.
Consider X_m is the mth row of the given image. X_m (n) is the nth pixel corresponding to the mth row of the original image? Similarly Y_m is the DCT out for mth row of the image. Y_m(n) is the nth DCT value corresponding to the mth row. Then the DCT value can be computed as follow:

$$Y_m = Y_{m-1}(\mathrm{n}), \text{ if } \mod(X_m(n) - X_{m-1}(n)) \leq 3$$

$$Y_m = Y_m(\mathrm{n}), \text{ Otherwise} \quad (3)$$

The equation (3) eliminates the DCT computation (Y_m) for a given row if the row (X_m) is matched with previous row (X_{m-1}) or the difference is less than or equal to 3. It eliminates 2 LSB bits per pixel and uses the 6 MSB bits for each pixel to compare the pixels in two rows. Similarly if we eliminate 3bits, 4 bits the threshold values for the difference is 7 and 15 respectively.

4. RESULTS AND DISCUSSIONS

For various input images the DCT values are obtained using the Comparative Input Method. After the DCT, original image is obtained using inverse DCT and then PSNR, MSE is calculated for the obtained images. This process is applied for different cases (considering the number of bits ignored to be 0, 1, 2, 3, 4) and the obtained images are shown in Figure-5 with the MSE and PSNR values. Different images with less and more intensity variations are considered for the computing DCT using our method. From the performance comparison in Figure-5 it shows that the output image is exactly same as the input image when '0' bits are ignored. When the number of bits ignored per pixel increases, the image quality also decreases slightly.

The comparison of the MSE and PSNR values obtained for each case of different input image is tabulated in Table-1. For all the input images, it shows that the MSE value increases rapidly when the number of bits ignored per pixel increases. And hence correspondingly the PSNR decreases as the number of bits ignored increases. A plot between the MSE values and the number of bits ignored per pixel corresponding to all the input images is depicted in Figure-6. The relationship between the number of bits ignored per pixel and the MSE is evident from the plot. The MSE values are insignificant till the number of bits ignored per pixel is 2. For 2 bit ignorance mandrill image offer lower MSE and Lena Image offer high (0.5805) MSE. When 3 bits are ignored, the MSE is considerable, whereas when the number of bits ignored becomes 4, the MSE value becomes significantly high. This is because in the last case, the acceptable difference between the two pixels (for row comparison) becomes 16. This causes a rapid change in the DCT output values.

Name	Original Image	N=0 bits	N=1 bit	N=2 bits	N=3 bits	N=4 bits
CAMERAMAN						
	MSE PSNR	0.000001 319.9627	0.0805 59.0737	0.4055 52.0514	1.9449 45.2419	11.1596 37.6543
LENA						
	MSE PSNR	0.000001 321.9418	0.1156 57.5003	0.5805 50.493	3.029 42.9586	16.1493 36.0493
PEPPERS						
	MSE PSNR	0.000001 322.0210	0.0486 61.2629	0.3784 52.3511	4.1263 41.9752	21.7505 34.7561

Figure-5. Performance comparison for various images obtained from the proposed DCT computation.

Table-1. Comparison of mse and psnr calculated for various images after number bits ignored for row comparison.

S. No.	Name of the image	N=0 bits		N=1 bit		N=2 bits		N=3 bits		N=4 bits	
		MSE	PSNR	MSE	PSNR	MSE	PSNR	MSE	PSNR	MSE	PSNR
1	Lena	0.000001	322	0.1	57.5	0.6	50.5	3	43	16.1	36
2	Cameraman	0.000001	320	0.1	59.1	0.4	52.1	1.9	45.2	11.2	37.7
3	Rice	0.000001	323	0.1	60	0.5	51.6	3.4	42.8	17.5	35.7
4	Mandrill	0.000001	321	0	62.3	0.2	55.2	2.2	47.7	20.3	35
5	Pirate	0.000001	322	0	62.7	0.2	54.4	2.3	44.6	17.3	35.8
6	Peppers	0.000001	322	0	61.3	0.4	52.4	4.1	42	21.8	34.8

Hence the best case is considered to be ignoring 3 bits i.e. considering a difference of 8 between the pixel values. For 3 bit ignorance MANDRILL image offer lower MSE (2.1792) and PEPPERS Image offer high (4.1263) MSE and PSNR of MANDRILL image is high (47.747).

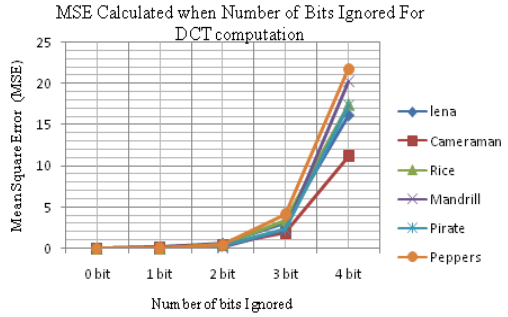

Figure-6. Comparison chart of MSE calculated for various images after number Bits ignored for Row comparison.

It also shows that in mandrill image pixels of adjacent row has more difference. It also shows that the CAMERAMAN image has range of the MSE values are relatively low, whereas for the PEPPERS the range is abruptly high in 4- bit comparison. Similarly, a plot between the PSNR values and the number of bits ignored corresponding to all the input images is depicted in Figure-7.

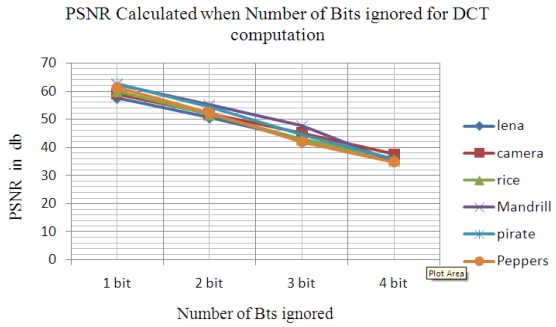

Figure-7. Comparison chart of PSNR Calculated for various images after number Bits ignored for Row comparison.

The inverse relationship between the PSNR and the number of bits ignored is evident. i.e. the image quality becomes degraded as the number of bits ignored increases, although the difference is not clearly perceivable in the output images. When the comparison is made between the various inputs, it can be seen that the PSNR for the CAMERAMAN image is high compared to other images for 4-bit elimination, hence a better output quality. The number of rows for which the DCT computation can be skipped is tabulated in Table-2. Comparing generally, it can be seen that for the CAMERAMAN image, the number of rows eliminated (2556) is much higher than the other images when 2 bits are ignored for row comparison, because much uniform background is present in the CAMERAMAN image. This is evident from the fact that even when the number of bits ignored is 0, more number of rows is similar reducing drastically the number of computations. But when the difference is relaxed to be 16 (for number of bits to be 4), much more number of rows becomes similar in some of the images compared to the others. This is decided by the range of the surrounding pixels for each row in the image. Here the higher number of eliminations is in LENA image. This is due to the fact that in the LENA image most of the pixel values lie in a similar range of intensities.

Table-2. Comparison of number of rows repeated for various images after number bits ignored for row comparison.

Image	N=0 bits	N=1 bit	N=2 bits	N=3 bits	N=4 bits
Lena	268	1255	2496	3911	5382
Cameraman	809	1868	2556	3213	4077
Rice	34	357	1304	2774	4310
Mandrill	21	122	495	1374	3220
Pirate	62	224	693	1653	3292
peppers	10	199	964	2811	4728

5. CONCLUSIONS

In this paper we have proposed a novel method for DCT computation for lossy image compression. 1D DCT computation is computed for a row is based on the difference between the pixel values of adjacent rows. In response to this method a larger number of computations are reduced when 5 bits and 4 bits of pixels are taken for comparison. The proposed method is verified with various high and less correlated images. The results show that image quality is maintained with minimum of 34.756 db PSNR and maximum of 37.54 db even though 4 bits are removed from 8 bits for a pixel for row comparison. Our method has reduced maximum 5382 computations out of 8192 and minimum 3220 computations out of 8192 when removing LSB 4 bits for Row comparisons.

REFERENCES

[1] N.Ahmed, T. Natarjan and K.R.Rao, "Discrete Cosine Transform", IEEE Transactions on Computer, vol. 23, no. 2, pp. 90-93 , 1974.

[2] S. Belkouch, M.El Aakif and A.Ait Ouahman, "Improved Implementation of a Modified Discrete Cosine Transform on Low-Cost FPGA", International Symposium on I/V Communications and Mobile Network. pp. 1-4, 2010.

[3] L.Byeong, "A New Algorithm to Compute the Discrete Cosine Transform", IEEE Transactions on Acoustics, Speech, and Processing, vol.32, no.6, pp.1243-1245, 1984.

[4] K.Keshab Parhi and Chao Cheng, "A Novel Systolic Array Structure for DCT", IEEE Transactions on Circuits and Systems-II, vol. 52, no. 7, pp. 366- 368, 2005.

[5] W.H.Chen, C.H. Smith and S.C.Fralick, "A fast computational algorithm for the discrete cosine Transform", IEEE Transaction on Communication, vol. 25, no. 9, pp. 1004-1009, 1977.

[6] M.El Aakif, S.Belkouch, N.Chabini and M.M.Hassani, "Low Power and Fast DCT Architecture Using Multiplier-Less Method", Proceedings of the International Conference on Faible Tension Faible Consommation, pp. 63-66, 2011.

[7] K.S. Geetha and M.Uttara Kumari, "A new multiplierless discrete cosine transform based on the Ramanujan ordered numbers for image coding", International Journal of Signal Processing, Image Processing and Pattern Recognition, vol. 3, no. 4, pp. 1- 14, 2010.

[8] G.Jyoti Chopra and Geetika, "Novel Image Compression Technique With Improved Wavelet Method", International Journal of Recent Technology and Engineering, vol. 1, no. 2, pp. 109-111, 2012.

[9] H. S.Hou, "A Fast Recursive Algorithm for Computing the Discrete Cosine Transform", IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 35, no.10, pp. 1455-1461, 1987.

[10] J.Hyeonuk, K.Jinsang and Won-KyungCho,"Low-power multiltiplierless DCT architecture using image correlation", IEEE Transactions on Consumer Electronics, vol. 50 , no .1 , pp. 262– 267, 2004.

[11] C.Loeffler, A.Lightenberg and G. S.Moschytz, "Practical fast 1-D DCT algorithms with 11 Multiplications", Proceedings of the International Conference on Acoustics, Speech and Signal Processing, vol.2, pp. 988-991, 1989.

[12] T.Prabhakar, V.Jagan Naveen, A.Lakshmi, Prasanthi and J.Vijaya Santhi, "Image Compression Using DCT and Wavelet Transformations", International Journal of Signal Processing, Image Processing and Pattern Recognition, Vol. 4, No. 3, pp.61-74, 2011.

[13] J.Rohit Kumar , "Design and FPGA Implementation of CORDIC-based 8- point 1D DCT Processor", E thesis, Department of Electronics and Communication Engineering, National Institute of Technology, Rourkela, Session , 2010.

[14] A.Shaofeng and C.Wang, "A computation structure for 2-D dct watermarking", IEEE international Midwest symposium on circuits and systems, pp. 577 - 580, 2009.

[15] v.k.sharma, k.k. mahapatra and c.umesh, "an efficient distributed arithmetic based vlsi architecture for dct", proceedings of the international conference on devices and communications, pp. 1-5, 2011.

[16] R.Sudhakar, S.Jayaraman, "Implementation of a Novel Efficient Multiwavelet Based Video Coding Algorithm", The International Arab Journal of Information Technology, vol. 5, no. 1, pp. 52-60,2008.

[17] M.Sukadev and Chandan Singh Rawat,"A Hybrid Image Compression Scheme using DCT and Fractal Image Compression", The International Arab Journal of Information Technology, vol. 10, no. 6, pp. 553-562, 2013.

[18] C.C. Sun, S.J. Ruan, B.Heyne and Goetze, "Low-power and high-quality Cordic-based Loeffler DCT for signal process", IET Transaction on Circuits, Devices & Systems, vol. 1, no.6, pp.453-461, 2007.

[19] M.Vashkevich and A.Petrovsky, "High-accuracy implementation of fast DCT algorithms based on algebraic integer encoding", International Conference on Signals and Electronic Systems. pp. 1-6, 2012.

[20] Vetterli M. and Nussbaumer H., "Simple FFT and DCT Algorithms with Reduced Number of Operations," Signal Processing (Elsevier), vol.6, no.4, pp. 267-268, 1984.