The DLMT hardware implementation. A comparative study with the DCT and the DWT

Felix Moreno and David Aledo
Centro de Electrónica Industrial (CEI)-ETSI-Universidad Politécnica de Madrid (Spain)

Abstract— **In the last recent years, with the popularity of image compression techniques, many architectures have been proposed. Those have been generally based on the Forward and Inverse Discrete Cosine Transform (FDCT, IDCT). Alternatively, compression schemes based on discrete "wavelets" transform (DWT), used, both, in JPEG2000 coding standard and in H264-SVC (Scalable Video Coding) standard, do not need to divide the image into non-overlapping blocks or macroblocks. This paper discusses the DLMT (*Discrete Lopez-Moreno Transform*) hardware implementation. It proposes a new scheme intermediate between the DCT and the DWT, comparing results of the most relevant proposed architectures for benchmarking. The DLMT can also be applied over a whole image, but this does not involve increasing computational complexity. FPGA implementation results show that the proposed DLMT has significant performance benefits and improvements comparing with the DCT and the DWT and consequently it is very suitable for implementation on WSN (*Wireless Sensor Network*) applications.**

I. INTRODUCTION

The reasons for the DCT [1], [2] success are based on the following factors: a) its computational simplicity, b) it is easy to be implemented in hardware and c) the relative quality of its results. However, its main drawback is the yielding of so-called "blocking artifacts" [3], [4], [5]. This effect is due to the implementation of the DCT over data blocks instead of applying it to the whole image (in the case of image compression standards 8x8pels blocks or 16x16pels macroblocks); where the correlation among neighboring blocks or macroblocks is not eliminated. On the other hand, the application of quantization operation (forward and inverse) is a data loss coding technique. This effect is particularly significant when high compression ratios and low bit-rate are needed.

The DWT [3],[6] transforms a time-domain discrete signal into time-frequency domain discrete signal. That is, by means the DWT a multiresolution analysis is achieved (time and frequency domains) and "blocking artifacts" do not arise, even though at high compression ratios.

Similarly the Fourier Transform decomposes a signal in a summation of sinusoids at different frequencies; the DWT represents a signal as a superposition of several "wavelets". These are small "waves" that have their power concentrated in certain time windows. Although there is large literature explaining the physical meaning of the DWT (continuous or discrete) based on "wavelets", it would be explained as follow, when decreasing the analysis time (scale) of the signal, then the resolution in the frequency domain signal increases. Conversely, the increasing of the analysis time (scale in the time domain) reduces the resolution in the frequency domain.

The major drawback of the DWT is its high computational cost, compared with the DCT [7], [8], [9], [10], [11]. On the other hand, systems based on DWT compression present their most effective performance when adapting the calculation algorithms to the type of signal (in the case of image compression, the type of the images and their spatial-temporal characteristics). Furthermore, taking into account the calculation algorithms which have been proposed in [11], based on the use of multiple stages of signal filtering, some blurring effects and ringing at the edges of the image would arise. The lifting scheme proposed by Sweldends [12], and used in [13] and in [14] reduces the computational cost of the transform but the need of adapting the calculation algorithms to the type of signal is still necessary.

The DLMT proposes a new scheme intermediate between the DCT and the DWT. Firstly, as discussed below, it is computationally very similar to the DCT and the quality of results obtained does not depend on the characteristics of the image being processed. Secondly, the DLMT uses quasi-sinusoidal functions, so the emergence of blocking artifact blocks and their effects have a relative low importance.

Finally, it is necessary to mention that the use of quasi-sinusoidal functions has allowed to achieve a multiresolution control quite close to that obtained by a DWT, but without increasing the computational complexity of the transformation. In addition, as shown later, the DLMT can also be applied over a whole image, but this does not involve increasing computational complexity. In this case, however, the necessary calculations are increased; but, because of its simplicity, it would be easily implemented in hardware.

II. THE DLMT

The two dimensional DLMT-2D consists of a pair of mathematical transforms, forward and reverse [16] (FDLMT-2D and IDLMT-2D).

Traditionally, complex exponentials have been used for the Fourier series development. In the same way, it is also possible to develop functions in sine or cosine series (base

sequences). This is the case of the DCT (Discrete Cosine Transform) and the DST (Discrete Sine Transform). According to [2] and [15] for a given function there are 8 ways to develop in the form of cosine series and 8 ways to develop in sine series. They constitute a family of 16 orthogonal transformations of real sequences, x(n).

Accepting an implicit or virtual periodicity of x(n) [15] and, also, an even symmetry, according to [2], the period of x(n) will be 2N, where N is the number of values that comprise the signal x(n) in a 2N time-domain virtual period. Thus, the pair of transforms (Forward and Inverse) DLMT-2D have been defined in [16].

The FDLMT-2D is quite similar to the FDCT-2D in the sense of the property of the low frequency energy compaction [15], [16]. However, the FDLMT-2D is much more efficient, as is discussed in [16]. This is due to the weighting functions are multiplied by their corresponding sine functions; thus *sinc* functions are obtained. They have very interesting properties both in the time-domain and in the frequency-domain.

As it is demonstrated in [16] it is easily deductible that zeros of the *sinc* function, and therefore zeros of the pair of transformed FDLMT-2D and IDLMT-2D, depend on the value of N. That is, the size of one of the two dimensions of the signal x(i,j), (8x8pels block, 16x16pels macroblock, etc). 2N is also the inherent or virtual periodicity of the signal to be processed, but also 4N, and 8N etc. This allows to control perfectly what it has been called scale (time-domain) of the signal (just at the introduction of this work), and its resolution in the frequency-domain. Since taking periods multiples of the inherent or virtual periodicity 2N or taking different values for N, it is possible to control the first zero of the *sinc* function. In other words, when increasing the value of N, the scale (time-domain) of the signal is also increasing; while the resolution (frequency-domain) is decreasing. Conversely, when decreasing the value of N the scale in the time-domain is decreasing while the resolution in the frequency-domain is increasing. This behavior is quite similar and the results are also comparable to those obtained by the DWT as discussed below. Also, in terms of pixel decorrelation and energy compaction when using *sinc* functions, in the frequency-domain, an explicit linear filter-bank is obtained with a variable and controllable cut-off frequency. This cut-off frequency depends on the inherent periodicity 2N and on the DLMT indexes "i" and "j". In summary, the proposed DLMT is biortogonal and multiresolution.

III. FDLMT-2D AND IDLMT-2D PROCESSING ALGORITHMS

In [16] the algorithm for calculating the FDLMT-2D and the IDLMT-2D that facilitates its implementation in hardware and software is proposed. Its study and its multi-resolution property demonstration will be done by a demonstrative hardware implementation example.

The solution proposed in [16] is simple and it is presented in the case of the DLMT-2D by means of the property of separation of the two dimensions, because it is a biortogonal transformation. Therefore, the DLMT-2D is separable, being able to express directly the FDLMT-2D as:

$$X(k,l) = [C][x(i,j)][C^T] \quad (1)$$

Where [C] is coefficient matrix and [C^T] is the transposed matrix.

On the other hand, the equation for calculating the inverse transformation can be derived:

$$x(i,j) = [C]^{-1}[X(k,l)][C^T]^{-1} \quad (2)$$

Those expressions are computationally very similar to those used traditionally (especially in hardware implementations) for the FDCT-2D and the IDCT-2D and much simpler than those used in the calculation of the DWT-2D. Regarding the FDLMT-2D, to take advantage of that algorithm, pixels must be received in 8x8 pels blocks, row by row and in order to be able to compare results.

The first row multiplications of the so-called post-matrix-multiplication (pomm=[x(i,j)].[C^T]) are calculated first for the 8 pels of the first row and then for the following first row of next block and so on until the end of the picture line (see Fig. 1).

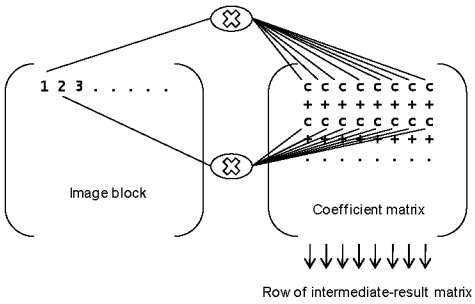


Fig. 1. Post-matrix-multiplication diagram (pomm=[x(i,j)].[C^T])

The second row of the first block is then calculated and so forth. This operation is carried out multiplying each incoming pels by all row coefficients, i.e. the first incoming pel is multiplied by the coefficients of the first row. The second incoming pel is multiplied by the coefficients of the second row, and so on. Then, those products are accumulated and a row of the intermediate-result matrix is obtained. These intermediate results are stored in a 2x8 data rows size memory. So, it is possible to read one of these rows while writing or updating the other.

Data is read in that way in order to carry out the so-called pre-matrix-multiplication (prmm = [C].[pomm]), which is very similar to pomm from the processing algorithm point of view (see Fig. 2).

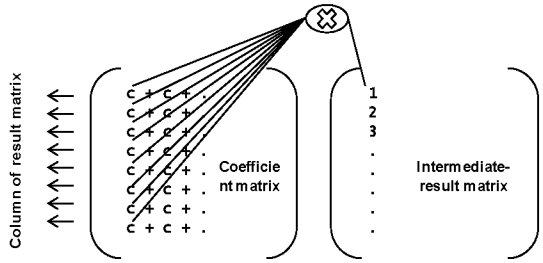


Fig. 2. Pre-matrix-multiplication diagram (prmm=[C].[pomm]).

Concerning the IDLMT-2D, the same processing algorithm is accomplished. However, in this case pre-matrix-multiplication is carried out first and then post-matrix-multiplication.

IV. HARDWARE ARCHITECTURE

Each matrix multiplication described in the above processing algorithm section is carried out by a hardware processing block. These hardware blocks consist of eight multipliers, eight accumulators and one shift register with parallel load. The input data are multiplied by each coefficient of the corresponding row or column. Activating an enable signal, products are accumulated. When eight enable activations occurs the accumulated results are loaded in parallel into de shift register and the accumulators are loaded with new products. At the same time, when enable signal is activated the shift register is shifted showing a new result at the output (see Fig. 3).

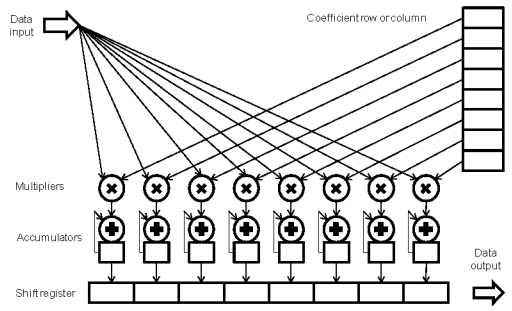


Fig. 3. Processing block diagram

The latency of those processing blocks therefore is only eight enable signal activations which, as discussed below, occur every four clock cycles. That is 4*8 = 32 clock cycles (in the tested system the clock was 50 MHz). The VHDL description of those processing blocks is:

```
entity GenPre is
  generic
  (
    NbitIn  : natural := 16; --Num.Bits of the input
    NbitC   : natural := 18; --Num.Bits of the coefficients
    NbitOut : natural := 16; --Num.Bits of the output
    NvlComp : natural := 4; --Range of compressed matrix
    IniCont : natural := 6; --Initial value of the counter
    ClipMax : natural := 29
  );

  port
  (
    -- Input ports
    reset : in std_logic;
    clk   : in std_logic;
    en    : in std_logic; --Accumulators enable
    en_r  : in std_logic; --Shift register enable
    data  : in  std_logic_vector(NbitIn-1 downto 0);
    Column_C : in  vector; -- Input coefficients
    Int   : in std_logic_vector(3 downto 0);

    -- Output ports
    count_C : out integer range 0 to 7;
    output  : out std_logic_vector(NbitOut-1 downto 0)
  );
end GenPre;
```

In order to avoid overflow problems, results of the multipliers have to have a length of *NbitIn+NbitC* bits. For de same reason, the length of the accumulated results have to be *NbitIn+NbitC+3*. But memories cannot store any length of words. In the tested system intermediate memories was 16 bits length per word and the final memory (for the entire transformed or recovered image) was 8 bits length per word. Moreover, coefficients are integer (positive or negative), and therefore accumulations can increment or decrement the final result.

Fig. 4 shows the final processing architecture implemented, where FDLMT-2D and IDLMT-2D are performed consecutively. This allows to check results on the display in real time. This architecture has been also used for implementing the DCT-2D (FDCT and IDCT) that has enabled a comprehensive comparison of the results obtained by both algorithms, DLMT and DCT.

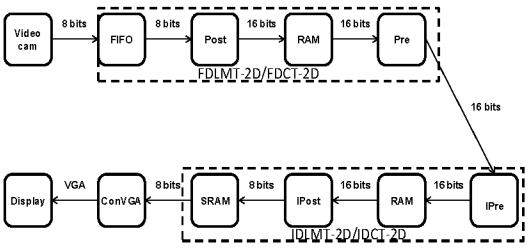


Fig. 4. Processing architecture. (RAM: FPGA internal memory). (SRAM: external memory).

The whole system control is synchronized with the video cam vertical and horizontal sync signal. The state machine designed has 4 states (1 state takes 1 clock cycle) which are repeated cyclically:

- Cycle 1: Input FIFO Reading and SRAM updating for displaying.
- Cycle 2: Pre and IPost blocks enable.
- Cycle 3: Internal RAM updating
- Cycle 4: SRAM Reading and VGA displaying.

V. FPGA IMPLEMENTATION RESULTS

The synthesis results obtained have shown that those FPGA with integrated hardware multiplier would be necessary in order to achieve real-time processing. The implemented algorithm (see [16]) is doing 32 multiplications by means of a typical pipelining architecture, which has a measured initial latency of 64 clock cycles. Therefore, using a 12.5MHz clock system, the elapsed time is 5.12µsec. The development board used is a DE2 from ALTERA, which contains a CICLONE II (EP2C35F672C6) FPGA device with 35 hardware 18x18bits multipliers. So, concerning resources consumption relevant data can be provided: Embedded multipliers (91%), logic elements (8%), memory (68%).

The prototype platform used in [13] and [14] however is an ML507 development board, which contains a Xilinx Virtex 5 (XCVFX70T) FPGA device with an embedded PowerPC processor, responsible of running the Evolutionary Algorithm described. In that implementation, one result is obtained each two cycles, so the computation time of the whole wavelet at 150MHz clock cycle is 0,873msec. Also, the mutation process takes 2 minutes and only still images were used in those experiments.

The DLMT then is a faster algorithm and at the same time better, or at least similar, results have been obtained. Besides, the DLMT does not need to be adapted to the image type as reported in [13] and [14]. In the same way, the DLMT improves the hardware cost because a medium size low-cost device can be used (e.g. Xilinx Spartan 3 family, ALTERA CICLONE II, etc).

The DLMT-2D algorithm implemented needs a little bit larger memory to store coefficients matrix and coefficients inverse matrix; however no quantization process is necessary at all, as in the case of the DCT-2D.

VI. ILLUSTRATIVE EXAMPLES

In the following, results obtained using a training set of 256x256pels grey-scale still image are presented (fingerprint. In [16] results obtained using lena, goldhill and baboon are also presented). Traditionally, the performance is evaluated by the PSNR (Peak Signal to Noise Ratio) [7], [9], [10] and the CR (Compression Ratio):

$$\mathrm{PSNR_{dB}} = 10\log_{10}\frac{\mathrm{Max.Val.}^2}{\mathrm{MSE}} \qquad CR = \frac{(Nboi - Nbci)x100}{Nboi} \tag{3}$$

Where *Max.Val.* is the maximum value of the input data (in the case of a luminance matrix it would be 235), *Nboi* is the Number of bits of the original image, *Nbci* is the number of bits of the compressed image and *MSE* is the Mean Square Error.

$$\mathrm{MSE} = \frac{1}{\mathrm{NxN}}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left(x(i,j) - x'(i,j)\right)^2 \tag{4}$$

Where *x(i,j)* is the input data matrix (8x8pels of luminance and 8bit per pel) and *x'(i,j)* is the processed data matrix.

All the results can be compared to the Evolutionary Algorithm (EA) proposed in [13] and [14]. Where the best fit PSNR for a set of 80 test images is 36,1490dB and an average fit PSNR of 29,8079dB after 500 mutations is obtained. Those experiments were carried out using a training set of 256x256pels grey-scale still image with a 2 or 3-level DWT. As a result of that, 128x128pels or 64x64pels images were obtained.

In the case of the DLMT depending on the image entropy and on the image block size (N=8 means 8x8pels block size. Some representative examples with 8x8pels, 16x16pels and the whole image 256x256pels are shown) compression rate from 68% to 75% can be achieved without any complex decimation algorithm which is really not needed. However, compressed image have been still scaled (here, scale has the same meaning than DWT level or decimation process) by 2 or even by 4, in order to get 128x128pels or 64x64pels images respectively and to be able to compare these results although there is not a noticeable improvement. In those cases the PSNR decreases even to 22,50dB which are yet comparable to the results obtained in [13] and [14]. In all those experiments a very simple interpolation algorithm has been employed for recovering the original image size, based on the duplication of the neighbor pel. Consequently, it is clear that when using a little bit more sophisticated interpolation algorithm (e.g. median filtering) the PSNR is increasing significantly.

It is clear that the scaling down process always harms the system performance. It is not really necessary because the DLMT itself reaches CR greater than 70% with a PSNR significantly higher than those supplied in [13] and [14]; even when applying over higher entropy images (e.g., baboon image) [16].

Fig. 5 shows a comparison between: a) original 256x256pels gray-scale fingerprint, b) recovered image after FDLMT-2D and IDLMT-2D processing with Scale=1. And no differences are noticed.

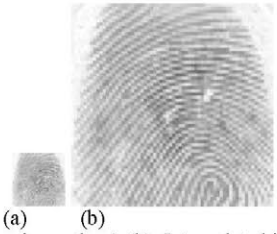
(a) (b)

Fig. 5. (a): Original image. (b): Recovered image

Fig. 6 and Fig. 7 show the FDLMT-2D and IDLMT-2D processed scaled-down image (by 2 and by 4) and the interpolated ones, where some artifacts can be noticed in Fig. 7-(b).

(a) (b)

Fig. 6. (a): Scaled-down image by 2. (b): Interpolated image



(a) (b)

Fig. 7. (a): Scaled-down image by 4. (b): Interpolated image

Finally, for a Lena 256x256pels gray-scale image the same experiments were carried out with similar results [16].

On the other hand, a 640x480pels (30fps) Toshiba TCM38230MD(A) video camera has been also used (not only still images). So, processing one pel per clock cycle, each frame takes 24.576μsec. As a result of that, the maximum processing time is 29.696μsec which is less than the inverse the video frame rate 1/30fps=33msec.

Taking into account that the maximum available bandwidth in one of the most widely used WSNs (Zigbee) [17] is 250Kbps and that the achieved FDLMT mean compression ratio is 75% [16].

TABLE I

Image size formats and bit-rates (Emphasizing those suitable for ZigBee WMSNs)

	Frame rate	Resolution	Bit-rate	Scaled by 2	Scaled by 4
VGA	30	640x480	3.7Mbps	1.85 Mbps	921.6 Kbps
VGA	15	640x480	1.85Mbps	921.6 Kbps	460.8 Kbps
QVGA	30	320x240	921.6Kbps	460.8Kbps	240Kbps
QVGA	15	320x240	460.8Kbps	240Kbps	120Kbps
CIF	30	352x288	1.2Mbps	610Kbps	305Kbps
CIF	15	352x288	610Kbps	305Kbps	153Kbps
QCIF	30	176x144	304.4Kbps	153Kbps	76Kbps
QCIF	15	176x144	152.2Kbps	76Kbps	38Kbps

Table 1 shows the image size formats able to be supported by that WMSN (*Wireless Multimedia Sensor Networks*) and the maximum bit rate (also when the pictures are scaling down by two or by four) when considering only monochromatic pictures (only luminance, Y, is processed).

At the present an efficient low-cost hardware video coding architecture based on the DLMT for WSN is being just as a proof of concept, so no noticeable effort was made in optimizing hardware system implementation. No dynamic video scalability (spatial, temporal or quality) is considered. Our approach consists on having only one single bit stream adapted in terms of maximum quality, frame rate, resolution and available bandwidth for each WSN implementation. Some experiments will be carried out soon using WMSN based on [18] and [19]. Only the temporal scalability would be modified dynamically depending on the temporal entropy (whenever two consecutives frames were significantly different: what is defined as inter-frame entropy).

VII. CONCLUSION

This paper describes a new image coding scheme based on the DLMT. Just from the analysis of the previous results it can be derived that the FDLMT-2D has a better low frequency energy compaction than that obtained by the DWT. That makes the operation of quantization unnecessary (one of the sources of losses in image coding), which is typical in those coding systems based on the FDCT-2D. Nevertheless, in the case of the FDLMT-2D, the coefficient matrix can be divided by a constant (DLMT quantization), which is not strictly a quantization to improve the compression factor when applying RLE (Run-Length Encoding) and Huffman encoding. Besides, it would be possible to remove (or, at least, to simplify) the typical algorithm for reading the information (Zig-zag Scan), which would further improve the processing speed due to the simplification of the process.

The DLMT used here attempts to exploit the effect of the processed block size (NxNpels) and/or the inherent periodicity of the FDLMT (See [16]). When increasing the value of N, the scale (time-domain) of the signal is also increasing while its resolution (frequency-domain) is decreasing (the cut-off frequency of the filter-bank decreases). So, when processing images with a high entropy (baboon and goldhill images [16]) the DLMT allows using smaller values of N (smaller scale, time-domain) and increasing resolution (frequency-domain) while maintaining good visual quality. At the same time it is even possible to increase the virtual periodicity (4N, 8N or 16N, etc) so; as a consequence of that, the CR approaches 75% and the PSNR exceeds 30dB, improving the energy compaction without using any lifting scheme or evolutionary scheme as in [13] and [14]. Finally, the experiments demonstrate that the DLMT is producing better results than the DCT when increasing block size and using a constant periodicity of 2N. This encouraging coding scheme is very suitable for implementation in hardware without a noticeable increase of

the computational load and consequently for implementation on WSN applications. Further results will be reported.

The current status of this work shows how a new approach to the compression schemes can be faced. No quantization operation, as in DCT-based image compression traditional schemes is necessary. There is not any image dependence, as in those based on DWT. No Evolutionary Algorithm is needed to achieve best results in terms of PSNR and computational load; even when an algorithm based on an optimized lifting scheme is used. The DLMT approach can be implemented on relative small FPGA devices at relative low clock frequencies which is a great advantage when faced low-power and low-cost solutions (e.g. H264-SVC based solutions).

AKNOWLEDGMENT

The author wishes to express his deep gratitude for advice he has received from Prof. J. Lopez of the Universidad de Las Palmas de Gran Canaria (Spain). He proposed the initial ideas and my appreciation for that can be envisaged in the DLMT acronyms. Special thanks go to Prof. Ramirez-Angulo of the New Mexico State University (NMSU-USA) for his proofreading efforts and for his support and advice during my sabbatical period at the NMSU in 2010.

REFERENCES

[1] K. R. Rao and P. Yip, Discrete Cosine Transform. Algorithms, Advantages, Applications, Academic Press Inc., 1990, ISBN 012-580-203X.

[2] Stephen A. Martucci, Symmetric Convolution and the Discrete Sine and Cosine Transform, IEEE Transactions on Signal Processing, Vol. 42, No. 5, pp. 1038-1051, May 1994.

[3] Marc Antonini, Michel Barlaud, Pierre Mathieu and Ingrid Daubechies, Image Coding Using Wavelet Transform, IEEE Transactions on Image Processing, Vol. 1, No. 2, pp. 205-220, April 1992.

[4] Jianmin Jiang and Guocan Feng, The Spatial Relationship of DCT Coefficients Between a Block and Its Sub-blocks, IEEE Transactions on Signal Processing, Vol. 50, No. 5, pp. 1160-1169, May 2002.

[5] A. N. Skodras, Direct Transform to Computation, IEEE Signal processing Letters, Vol. 6, pp. 202-204, Aug. 1999.

[6] S. Mallat, A Wavelet Tour of Signal processing, second edition, Academic Press, Sep. 1999.

[7] V. Srinivasa Rao, Dr. P. R. Kumar, G. V. H. Prasad, M. Prema Kumar and S. Ravichand, Discrete Cosine Transform Vs Discrete Wavelet Transform: An Objective Comparison of Image Compression techniques for JPEG Encoder, International Journal of Advanced Engineering and Applications, pp. 87-90, Jan. 2010.

[8] Zixiang Xiong, Kannan Ramchandran, Michael T. Orchard, and Ya-Qin Zhang, A Comparative Study of DCT- and Wavelet-Based Image Coding, IEEE Transactions on Circuits and Systems for Video Technology, Vol. 9, No. 5, pp. 692-695, Aug. 1999.

[9] Ali Al-Haj, Combined DWT-DCT Digital Image Watermarking, Journal of Computer Science 3 (9), pp. 740-746, 2007. ISSN 1549-3636.

[10] Wen-Chien Yan and Yen-Yu Chen, DCT-Based Image Compression Using Wavelet-Based Algorithm with Efficient Deblocking Filter, 14th European Signal Processing Conference (EUSIPCO 2006), Florence, Italy, Sept. 4-8, 2006.

[11] Hznnu Olkkonen and Juuso T. Olkkonen, Discrete Wavelet Transform Structures for VLSI Architecture Design, InTech Publ. 2010, ISBN: 978-953-049-0.

[12] W. Sweldends, The lifting scheme: A custom-design construction of biorthogonal wavelets, Appl. Comput. Harmon. Anal., Vol. 3, no. 2, pp. 186-200, 1996.

[13] R. Salvador, F. Moreno, T. Riesgo and L. Sekanina, Evolutionary approach to improve wavelet transforms for image compression in embedded systems. EURASIP, Journal on Advances in Signal Processing. Hindawi Publishing Co. (pp. Vol. 2011, Dec. 2010). ISSN: 1687-6172. e-ISSN: 1687-6180.

[14] Rubén Salvador, Alberto Vidal, Félix Moreno, Teresa Riesgo, Lukas Sekanina, Bio-inspired FPGA Architecture for Self-Calibration of an Image Compression Core based on Wavelet Transforms in Embedded Systems, SPIE 2011. 18-20 April 2011.

[15] Alan V. Oppenheim, Roland W. Schafer and John R. Buck, Discrete-Time Signal Processing, 2nd Edition, Ed. Prentice Hall, 2008, ISBN 978-84-205-2987-5.

[16] Félix Moreno and David Aledo, The DLMT. An alternative to the DCT. IECON 2011 Annual Conference of the IEEE Industrial Electronics Society. 7-10 Nov. 2011, Melbourne, Australia. Pp. 2267-2272, ISBN 978-1-61284-971-3.

[17] Harjito, B.; Han. S. Wireless multimedia sensor networks applications and security challenges. In Proceedings of the International Conference on Broadband, Wireless Computing, Communication and Applications, Fukuoka, Japan, 4–6 November 2010; pp. 842–846.

[18] Jorge Portilla, Ángel de Castro, Eduardo de la Torre and Teresa Riesgo, A Modular Architecture for Nodes in Wireless Sensor Networks. Journal of Universal Computer Science, vol. 12, no. 3 (2006), 328-339.

[19] Juan Valverde, Andrés Otero, Miguel Lopez, Jorge Portilla, Eduardo de la Torre and Teresa Riesgo. Using SRAM Based FPGAs for Power-Aware High Performance Wireless Sensor Networks. Sensors 2012, 12, 2667-2692; doi:10.3390/s120302667