Fast Multiplication-free QWDCT for DV Coding Standard

Antonio Silva, Paulo Gouveia, and Antonio Navarro, *Member, IEEE*

Abstract — *This paper deals with fast computation of some operations included in the digital video (DV) coding standard. The proposed solution converts floating-point arithmetic operations into integer arithmetic operations, replacing highly computational demanding blocks such as discrete cosine transform (DCT), weighting (W) and quantization (Q) by fast integer calculations using only shifts and additions. The overall computational complexity was reduced by 73% in comparison to a floating-point implementation. Our solution is suitable to be programmed into any fixed-point arithmetic processor decreasing the consumer equipment cost. This solution is still compatible with the standard in terms of the DCT precision requirements*[1].

Index Terms — Digital Video, Discrete Cosine Transform, Fixed-point Processing, Quantization.

I. INTRODUCTION

In the last decade, several compression algorithms have been standardized by international organizations like ISO, ITU and IEC. Video recording requires high picture quality and encoding schemes allowing random access. In majority, professional digital recording technologies have been defined by private companies. However, it evolved from the professional side like D-series and Betacam to the consumer market by the introduction of the IEC DV Standard [1], [2] and some of its variations, DVCAM [3] and DVCPRO. Rich video contents can then be obtained from the consumers, saved on servers and shared by millions of users all over the world. The DV standard was actually developed to be a high-performance successor of the existing consumer analogue formats (VHS and Hi-8) for video/TV recording in mini tapes and is emerging as a popular alternative in digital video storage. It has a compression ratio of 3:1 to 5:1 and it is suitable for devices such as digital camcorders, VCRs and video editors. The introduction of the DV standard was mainly motivated by the need of small size digital camcorders with some constrains such as recording mechanism size, cassette size, power consumption and consumer price.

The DV standard is a coding system with fixed bit rate of 25 Mbit/s (compression ratio of 5:1), uses intraframe compression and uses the DCT to remove redundancy from pixel block data. Once the DCT is computed, the coefficients are quantized and entropy encoded (Variable Length Coding, VLC). In order to have a fixed compressed data rate of 25 Mbit/s, DV uses a feed-forward video compression scheme which consists of selecting, according to the "activity" of the DCT block, the appropriate quantization table/step, conducting after entropy coding to a data stream close to the ideal fixed data rate. Being the central part of many image coding applications, all DCT based video algorithms or standards will benefit from a DCT fast computation.

Several floating-point DCT calculation algorithms have been proposed, and usually can be classified into two classes: indirect and direct methods. The former computes the DCT through a FFT or other transforms and the latter through matrix factorization or recursive computation.

When direct methods are chosen to calculate (NxN)-point 2-D DCTs, the conventional approach follows the row-column method which requires 2N sets of N-point 1-D DCTs. In [4], [5], the authors propose two 2-D DCT recursive algorithms based on fast 1-D DCT algorithms of [6], [7]. However, true 2-D techniques are more efficient than the conventional row-column approach. A direct 2-D method for the 2-D DCT based on polynomial transform techniques was provided by Duhamel and Guillemot [8]. Feig and Winograd [9] present a matrix factorization algorithm of 2-D DCT matrix. In [10], Vetterli propose an indirect method to calculate 2-D DCT by mapping it into a 2-D DFT plus a number of rotations. The 2-D DFT was computed through polynomial transform techniques. From a literature review, as far as we know the fastest 2-D DCT calculation [11] is due to Feig-Winograd's algorithm mentioned above.

Given an image with integer intensity values, the DCT transforms them into floating-point numbers (DCT coefficients), whose computational complexity can not be neglected. Efficient implementation of the DCT requires fixed-point implementations resulting in less silicon area and power consumption. However, in fixed-point implementation, there is an inherent accuracy problem due to finite word length.

In this paper we propose a joint implementation of the blocks, DCT, weighting and quantization (QWDCT) with the advantage of reducing considerable the computational complexity associated with these operations. For comparison purposes, we developed a DV reference software. Those blocks have a computational complexity of 38% relatively to the complete DV coding algorithm. With our QWDCT implementation, the overall computational complexity of the DV reference encoder was reduced of 27%. Explicit quantization of the DCT coefficients is avoided including the quantization values into the DCT computation.

We propose an integer multiplication-free algorithm to compute the QWDCT through the replacement of multiplications by shifts and additions. In order to reduce the number of operations needed to perform the DCT algorithm, the multiplicative values are approximated with several precisions. These approximations reduce the precision of the DCT, still maintaining compatibility with the standard specifications and resulting in a negligible subjective and objective video degradation.

---

[1] The authors are with the Telecommunications Institute, University of Aveiro, 3810-Aveiro, Portugal (e-mail: navarro@av.it.pt).

Contributed Paper
Manuscript received November 11, 2003

The paper is organized as follows. Section II provides a brief description of the DV coding system. In Section III, we analyze Feig-Winograd's algorithm and propose an integer multiplication-free QWDCT algorithm. Section IV presents simulation results to verify DV quality conditions and objective reconstructed video quality. Finally, conclusion remarks are presented in Section V.

II. DV CODING SYSTEM

The DV standard uses two different cassettes, the standard cassette (4.9x3x0.57 inch) designed to use in DV video recorders, and the small cassette (2x2.2x0.5 inch) intended for DV camcorders. It specifies a metal evaporate tape width of ¼ inch (6.35 mm) to record high quality digital video. Based on DV standard, professional systems emerged like DVCAM and DVCPRO. The basic video encoding algorithm for the professional systems are very similar to DV and the main differences are on the mechanical specifications of the tape.

In DV video coding system, as depicted in Fig. 1, the input video frame is organized into Macroblocks (MBs), each one with 4 luminance blocks and 2 chrominance blocks Cb and Cr. The image sampling procedure is performed according to ITU-R Rec.601 [12] with 720 pixels per line for both 525/60 and 625/50 systems. For the chrominance signals in the 525/60 system, it is performed a 4:1:1 horizontal subsampling, with both Cb and Cr signals sampled at 3.375 MHz (180 pixels per line), whereas for the 625/60 system is performed a 4:2:0 vertical subsampling by a factor of 2 with both chrominance signals sampled at 6.75 MHz (360 pixels per line). In 525/60 system, a macroblock (MB) is formed by 4 horizontally adjacent luminance blocks (4x1) and 2 chrominance blocks whereas in 625/50 system, the luminance blocks are also vertically adjacent forming 2x2 block matrix.

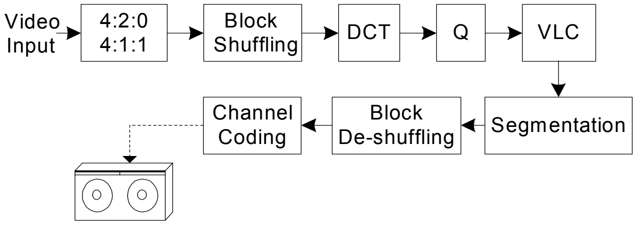


Fig. 1. DV encoding system.

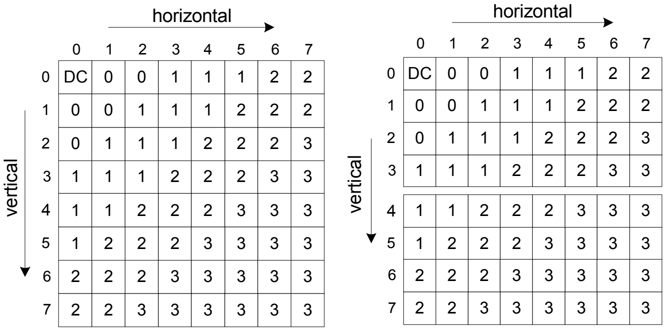
The video data is handled on a MB basis and the compression process is applied over 5 MBs taken from different frame regions. Generally speaking, this shuffling process ensures an adequate bit rate distribution over those frame regions, allocating less bit rate to the frame edges in order to achieve a good subjective quality picture at 25 Mbit/s. Since intraframe based compression is used, and due to the interlaced nature of a picture, there are two DCT modes in DV standard to overcome the problem of picture movement in a frame. Instead of using an 8x8 DCT, a 4x8 DCT is applied to each field data. Once the DCT is computed, DCT coefficients are weighted by a matrix of floating-point values. The weighting process essentially reduces DCT coefficients to zero. Higher frequency coefficients are more reduced than lower frequency coefficients. Since the weighting matrix was determined from studies on visual human perception, reduction will result in image unnoticeable effects. However, the DV standard specification defines the precision of the WDCT algorithm. The W matrix elements for 8x8 and 2x4x8 DCT are, respectively, given by,

$$W(i,j)=\begin{cases}1/4 & \text{for } i=j=0\\ \dfrac{w(i)w(j)}{2} & \text{otherwise}\end{cases}$$

$$W(i,j)=\begin{cases}1/4 & \text{for } i=j=0\\ \dfrac{w(i)w(2j)}{2} & \text{for } j<4\\ \dfrac{w(i)w(2(j-4))}{2} & \text{otherwise}\end{cases} \tag{1}$$

where w(0) = 1; w(1) = CS(4)/(4xCS((7)xCS(2)); w(2) = CS(4)/(2xCS(6)); w(3)=1/(2xCS(5)) w(4) = 7/8; w(5) = CS(4)/CS(3); w(6) = CS(4)/CS(2); w(7) = CS(4)/CS(1) and CS(i) = cos(iπ/16).

Further data compression takes place by quantizing the weighted AC coefficients and then entropy coding (Huffman VLC technique). As referred above, one can have two types of DCT blocks, 1x8x8 or 2x4x8, and a set of quantization matrices. As shown in Fig. 2, each matrix is divided into four areas and each of them has an associated quantization step.



	0	1	2	3	4	5	6	7
0	DC	0	0	1	1	1	2	2
1	0	0	1	1	1	2	2	2
2	0	1	1	1	2	2	2	3
3	1	1	1	2	2	2	3	3
4	1	1	2	2	2	3	3	3
5	1	2	2	2	3	3	3	3
6	2	2	2	3	3	3	3	3
7	2	2	3	3	3	3	3	3

(a)

	0	1	2	3	4	5	6	7
0	DC	0	0	1	1	1	2	2
1	0	0	1	1	1	2	2	2
2	0	1	1	1	2	2	2	3
3	1	1	1	2	2	2	3	3
4	1	1	2	2	2	3	3	3
5	1	2	2	2	3	3	3	3
6	2	2	2	3	3	3	3	3
7	2	2	3	3	3	3	3	3

(b)

Fig. 2. Quantization matrix organization (a) 8x8; (b) 2x4x8.

The DV specification defines a set of quantization tables, and the encoder selects the appropriate table to quantize each MB within a segment. Furthermore, each table is composed by a set of quantization matrices. DV has four quantization tables and 12 different possibilities of quantization matrices. The quantization steps associated to each area are shown in Table I. The DC coefficient is not quantized. Table I shows four different quantization tables (classes) and their corresponding four region steps (areas).

TABLE I
QUANTIZATION STEPS

Class N°				Area N°			
0	**1**	**2**	**3***	**0**	**1**	**2**	**3**
15				1	1	1	1
14				1	1	1	1
13				1	1	1	1
12	15			1	1	1	1
11	14			1	1	1	1
10	13		15	1	1	1	1
9	12	15	14	1	1	1	1
8	11	14	13	1	1	1	2
7	10	13	12	1	1	2	2
6	9	12	11	1	1	2	2
5	8	11	10	1	2	2	4
4	7	10	9	1	2	2	4
3	6	9	8	2	2	4	4
2	5	8	7	2	2	4	4
1	4	7	6	2	4	4	8
0	3	6	5	2	4	4	8
	2	5	4	4	4	8	8
	1	4	3	4	4	8	8
	0	3	2	4	8	8	16
		2	1	4	8	8	16
		1	0	8	8	16	16
		0		8	8	16	16

* In class 3, all quantization steps are multiplied by 2

The class number and quantization index varying from 0 to 15 are decided by estimating the activity (block energy) of each MB to ensure a total bit rate close to 25 Mbit/s.

As shown in Fig. 1, the last processing blocks in DV standard are the Segment Framer, Deshuffling and Channel Encoder. The former limits a total of 2560 bits in each segment corresponding to 5 encoded MBs, ensuring a compressed video bit rate of 25 Mbit/s. Finally, the MBs are deshuffled, channel encoded and stored into the tape. Each frame fills up either 10 or 12 tracks depending on the video system, 525 or 625 lines, respectively.

III. INTEGER QWDCT SOLUTION

In DV standard, as the DCT coefficients are weighted and quantized, then there is an advantage, in terms of computational complexity reduction, in dividing the DCT operation into two separate operations, Scaled DCT (SDCT) and Scaling (S). Therefore, we compute a scaled version of the DCT, i.e. a proportional version of the DCT, and then include the scaling factors in the following stage. This stage is composed of S, W and Q blocks. Thus, our proposal implements those operations DCT, W and Q into two new separate blocks. The associated computational complexity is reduced mainly by performing the calculations in fixed-point arithmetic without multiplications.

We make use of the most promising SDCT algorithm which was developed by Feig and Winograd [10]. Section A provides a brief description of this algorithm.

A. Feig-Winograd's Algorithm

The one dimensional (1-D) DCT can be described as a matricial operation given by ,

$$Y = AX \tag{2}$$

where X is a 8-point pixel vector. The 2-D DCT is described by

$$Y = A(X)A^{T} = (A \otimes A)X \tag{3}$$

where X is an 8x8 pixel block, Y is the DCT coefficients matrix, ⊗ is the Kronecker or tensor product and A is the DCT matrix with elements given by,

$$\begin{aligned}&[A]_{ij} = a_{ij} = K_i \times \cos\left[\frac{\pi}{N}\left(j-\frac{1}{2}\right)(i-1)\right]\\&i, j = 1, \ldots, N\end{aligned} \tag{4}$$

$$\text{where} \quad K_i = \begin{cases}\sqrt{\dfrac{1}{N}} & \text{for } i = 1\\[2ex] \sqrt{\dfrac{2}{N}} & \text{for } i > 1\end{cases}$$

Feig and Winograd proposed an algorithm which essentially consists of DCT matrix factorization. According to [9], the DCT matrix A can be represented as a matricial product given by,

$$A = C_8 = P_8 D_8 R_{81} M_8 R_{82} \tag{5}$$

where D_8 is a diagonal matrix whose diagonal elements are {1; 0.3536; 0.1913; 0.4619; 0.2778; 0.4904; 0.4157; 0.0975}. M_8 is also composed of real values $\gamma(k)=\cos(2\pi k/32)$. $R_{82}=B_1B_2B_3$, P_8 is a permutation matrix and B_1, B_2, B_3, R_{81} and M_8 are given by,

$$R_{81}=\begin{bmatrix}1&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0\\0&0&1&1&0&0&0&0\\0&0&-1&1&0&0&0&0\\0&0&0&0&1&1&-1&0\\0&0&0&0&-1&1&0&1\\0&0&0&0&-1&-1&-1&0\\0&0&0&0&1&-1&0&1\end{bmatrix}, B_1=\begin{bmatrix}1&1&0&0&0&0&0&0\\1&-1&0&0&0&0&0&0\\0&0&0&1&0&0&0&0\\0&0&1&-1&0&0&0&0\\0&0&0&0&0&0&-1&0\\0&0&0&0&-1&0&0&1\\0&0&0&0&1&0&-1&0\\0&0&0&0&0&1&0&1\end{bmatrix}, B_2=\begin{bmatrix}1&0&0&1&0&0&0&0\\0&1&1&0&0&0&0&0\\1&0&0&-1&0&0&0&0\\0&1&-1&0&0&0&0&0\\0&0&0&0&1&0&0&0\\0&0&0&0&0&1&0&0\\0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&1\end{bmatrix},$$

$$B_1=\begin{bmatrix}1&0&0&0&0&0&0&1\\0&1&0&0&0&0&1&0\\0&0&1&0&0&1&0&0\\0&0&0&1&1&0&0&0\\1&0&0&0&0&0&0&-1\\0&1&0&0&0&0&-1&0\\0&0&1&0&0&-1&0&0\\0&0&0&1&-1&0&0&0\end{bmatrix}, M_8=\begin{bmatrix}1&&&&&&&\\&1&&&&&&\\&&1&&&&&\\&&&\gamma(4)&&&&\\&&&&1&&&\\&&&&&\gamma(4)&&\\&&&&&&\gamma(6)&\gamma(2)\\&&&&&&-\gamma(2)&\gamma(6)\end{bmatrix}, P_8=\begin{bmatrix}1&0&0&0&0&0&0&0\\0&0&0&0&-1&0&0&0\\0&0&1&0&0&0&0&0\\0&0&0&0&0&-1&0&0\\0&1&0&0&0&0&0&0\\0&0&0&0&0&0&0&-1\\0&0&0&1&0&0&0&0\\0&0&0&0&0&0&1&0\end{bmatrix}. \quad (6)$$

C_8X can be calculated by first computing the product $R_{81}M_8R_{82}$, which involves 5 multiplications and 28 additions, and then incorporating P_8D_8 into the post-processing stages. Since P_8 is a signed permutation matrix and D_8 is a diagonal matrix, the introduction of P_8D_8 into scaling is simply a pointwise multiplication.

The computation of the 2-D DCT on 8x8 points involves the product of the matrix $C_8 \otimes C_8$ given by,

$$(P_8D_8R_{81}M_8R_{82}) \otimes (P_8D_8R_{81}M_8R_{82}) \quad (7)$$

with a 64-pixel vector X. A standard result about tensor products allows us to rearrange (7) into

$$((P_8D_8) \otimes (P_8D_8))((R_{81}M_8R_{82}) \otimes (R_{81}M_8R_{82})) \quad (8)$$

From (8), we observe that the 2-D SDCT can be computed by first computing the product of the right term,

$$\begin{aligned}&\left(R_{81}M_8R_{82}\right) \otimes \left(R_{81}M_8R_{82}\right) = \\ &\left(R_{81} \otimes R_{81}\right)\left(M_8 \otimes M_8\right)\left(R_{82} \otimes R_{82}\right)\end{aligned} \quad (9)$$

with X and then incorporating the Scaling factor,

$$(P_8D_8) \otimes (P_8D_8) = (P_8 \otimes P_8) \otimes (D_8 \otimes D_8) \quad (10)$$

into the following processing stage. This strategy is advantageous because (10) is a product of a diagonal matrix and a signed permutation matrix.

To perform the 2-D SDCT, the multiplications by $R_{81} \otimes R_{81}$ and $R_{82} \otimes R_{82}$ are calculated using the row-column method in a total of 128 and 288 additions, respectively. The product by $M_8 \otimes M_8$ involves 54 non-trivial multiplications (where non-trivial multiplication means multiplications by non-integer values), 46 additions and 6 shifts. All together, the computation of a 2-D SDCT involves 54 multiplications, 462 additions and 6 shifts. A more detailed description about the algorithm can be found in [9].

The DCT computation given by (3) is then expressed as,

$$Y = C_8 \otimes C_8X \quad (11)$$

where X is a 64-point vector and $C_8 \otimes C_8$ is given by (8).

The general structure of Feig-Winograd's algorithm consists in: 1- a preaddition stage; 2- a multiplication stage, with non-trivial multiplications and 3- post addition stage. The only matrices that have non-integer values are D_8 and M_8. In the following sections, we will investigate the best combination of precisions of the approximated matrices M_8 and D_8. They will be part of SDCT and SWQ blocks, respectively.

B. *Integer QWDCT implementation*

Herein we present a fast algorithm to implement the WDCT and Q stages described in DV standard. As mentioned in Section I, this implementation is justified since the DCT, W and Q steps are three highly demanding blocks, and the algorithm proposed is an attempt to reduce the number of operations needed to perform these three operations. This is achieved by calculating a SDCT and by eliminating the explicit quantization of DCT coefficients through the integration of scaling, weighting and quantization into only one integer operation. Since all practical implementations are done in fixed-point arithmetic, the algorithm proposed is a QWDCT integer implementation.

In order to implement a multiplication-free computation of the DCT, non-integer values of matrix M_8 are approximated by a sum of powers of 2, where multiplications are then replaced by shifts and additions. Obviously, the approximation on M_8

will introduce errors in the DCT computation, and matrix D_8 which is included into weighting stage, is modified to minimize them.

Once M_8 elements are replaced by a sum of powers of 2, the objective is, from (5), to determine the diagonal matrix $\hat{D}_8$, in order to minimize the error introduced by the approximation of matrix M_8. $\hat{D}_8$ is calculated as

$$\hat{D}_8 = \left(P_8^{-1}C_8R_{82}^{-1}\hat{M}_8^{-1}R_{81}^{-1}\right).*I \tag{12}$$

where (.*) operation denote a pointwise multiplication.

Several approximations with decreasing precision were made to matrix M_8 and the corresponding matrix $\hat{D}_8$ was determined for each case. For every pair $\hat{M}_8$ and $\hat{D}_8$, tests were conducted in order to verify the DCT accuracy, and the results are shown in the next section.

After computation of the SDCT, the coefficients are weighted by a matrix including the weighting factors defined in DV standard and the scaling factors. Therefore, coefficients of the SDCT are weighted by the matrix $S\hat{W} = \hat{S}.*W$, where $\hat{S}$ is the scaling matrix which includes the scaling factors given by $(P_8 \otimes P_8)(\hat{D}_8 \otimes \hat{D}_8)$ and W is the DV weighting matrix given by (1). Therefore, denoting $\hat{Y}$ as the SDCT coefficient matrix, the computation of WDCT can be expressed as

$$\hat{Y}.*SW = \left[\left((R_{81}\otimes R_{81})(M_8\otimes M_8)(R_{82}\otimes R_{82})\right)X\right].*SW \tag{13}$$

where X is a 64-point vector formed by interlacing the columns of the 8x8 pixels matrix.

As all operations are carried out in the integer domain and in order to avoid precision loss, all calculations should be kept in the range of processor register length. Therefore, input pixel values X, in the range between 0 and 255, are scaled up by a constant 2^b, where b a positive integer. After computing (13) and as WDCT coefficients have 10 bit length [1], [2], then these coefficients are placed between b and b+10 bit positions.

Let us now focus on determining the best processor register length, L. Analyzing the matrices involved in the WDCT calculation, the minimum register length should be b+14 bits, because the dynamic range of $\hat{Y}$ is increased by 6 bits. As we will discuss in the next section, the matrix $M_8 \otimes M_8$ may lead to at most a right shift of 14 bits. Therefore, minimum value for b is 14 resulting in L=28. Nevertheless, less precision may be adopted for that matrix and consequently, it will require a corresponding less value for b and L.

As the quantization values in DV are powers of 2, quantization can be incorporated into the last stage without any extra operations by combining the quantization value with the $S\hat{W}$ matrix. Therefore, as the DV standard defines 12 different quantization matrices, we will get the same number of different $S\hat{W}Q$ matrices. In order to improve precision, after SWQ calculation, a rounding operation should be accomplished by adding 2^{b-1}.

IV. COMPUTATIONAL COMPLEXITY

The DV standard defines some tolerances that the DCT implementation must satisfy in order to maintain its accuracy and consequently an acceptable reconstructed video quality. The method of measuring the DCT operation precision is described in the Fig. 3.

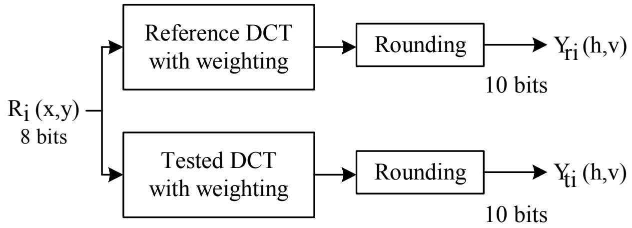


Fig. 3. DV accuracy test.

Data $R_i(x,y)$ is sent to both the reference DCT and integer DCT and the error between the two rounded DCT outputs, $Y_{ri}(x,y)$ and $Y_{ti}(x,y)$ is calculated. The input data $R_i(x,y)$ is composed of 100000 blocks with values generated from a uniform probability distribution in the range between -128 to 127. The reference DCT with weighting is calculated with double floating-point precision.

The measured error should satisfy the following criteria:

1) Probability of occurrence of error which is greater than 1 is less than or equal to $1x10^{-5}$.

$$P_r(|Y_{ti}(h,v) - Y_{ri}(h,v)| > 1) \le 1\times10^{-5} \tag{14}$$

where i = 0, 1, …, 9999, h = 0, 1, …, 7 is the horizontal coordinate and v = 0, 1, …, 7 is the vertical coordinate of the DCT coefficient block.

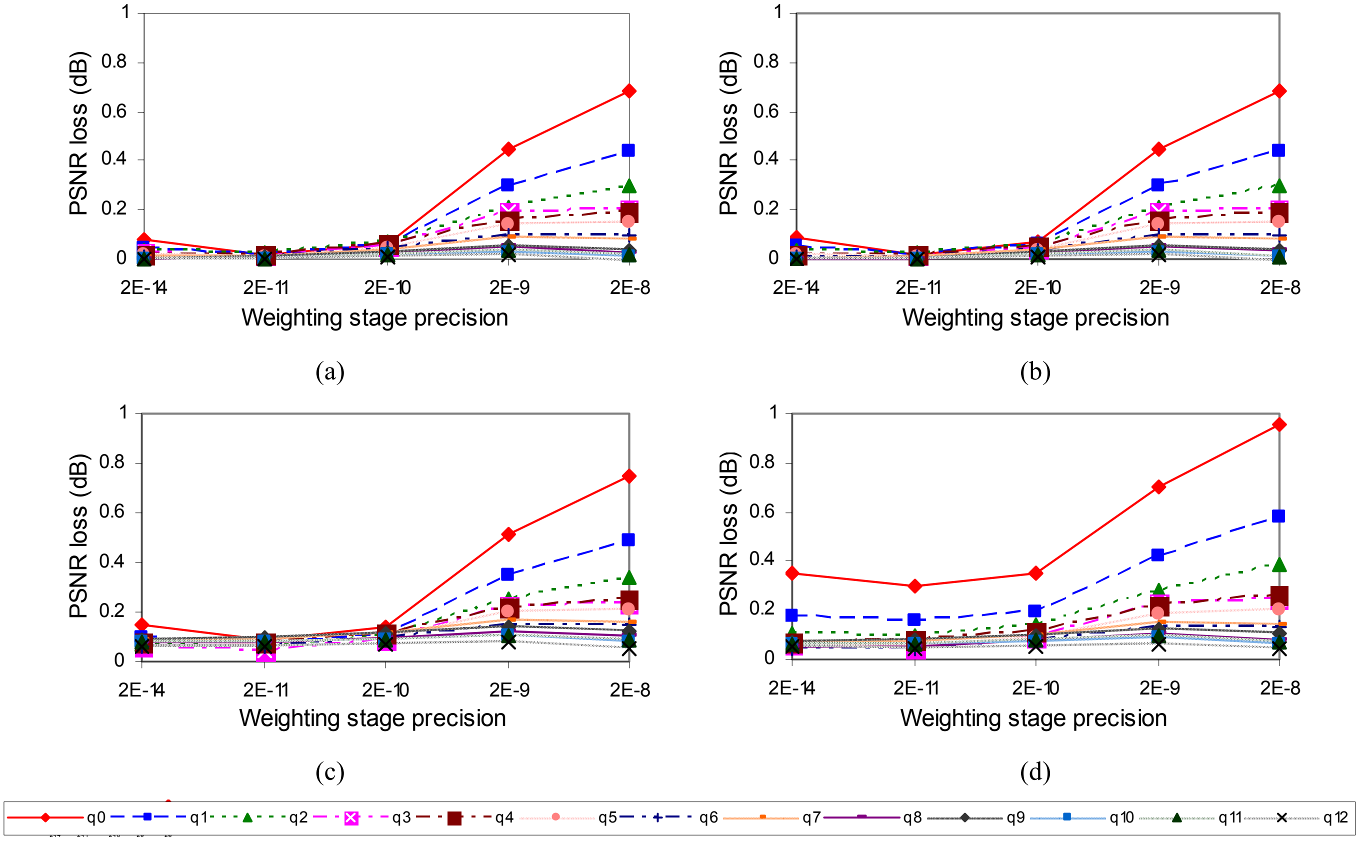
2) Mean square errors for all coefficients is less than or equal to 0.125.

$$\sum_{i=0}^{99999}\sum_{h=0}^{7}\sum_{v=0}^{7}\frac{\left(Y_{ti}(h,v) - Y_{ri}(h,v)\right)^2}{(64\times100000)} \le 0.125 \tag{15}$$

3) Maximum value of mean square error of each DCT block is less than or equal to 0.33.

$$\sum_{h=0}^{7}\sum_{v=0}^{7}\frac{\left(Y_{ti}(h,v) - Y_{ri}(h,v)\right)^2}{64} \le 0.33 \tag{16}$$

4) If all input pixel values of a DCT block are equal, all AC coefficients of the DCT block should be zero.



(a)

(b)

(c)

(d)

Fig. 4. PSNR loss between the reference WDCT and each WDCT implementation with different SDCT precisions, (a) 2^{-10}, (b) 2^{-9}, (c) 2^{-8}, (d) 2^{-7}.

Having established the conditions, and regardless of quantization, tests were performed to observe the behavior of our proposed integer multiplication-free WDCT implementation. As mentioned in Section III, the non-trivial multiplications were approximated in order to transform them into shifts and additions. The more precision is used, the more number of operations is required to perform the WDCT. In order to have trivial multiplications needed by the SDCT operation, the $M_8 \otimes M_8$ matrix elements were approximated to a maximum error of $1x10^{-4}$,

$$\begin{aligned}
&\gamma(2) = \cos(4\pi/32) = 1 - 2^{-4} - 2^{-6} + 2^{-9} + 2^{-14}\\
&\gamma(4) = \cos(8\pi/32) = 1 - 2^{-2} - 2^{-5} - 2^{-7} - 2^{-8} + 2^{-14}\\
&\gamma(6) = \cos(12\pi/32) = 2^{-2} + 2^{-3} + 2^{-7} - 2^{-13}\\
&\gamma(4)\gamma(2) = 2^{-1} + 2^{-3} + 2^{-5} - 2^{-9} - 2^{-10}\\
&\gamma(6) + \gamma(2) = 1 + 2^{-2} + 2^{-4} - 2^{-8} - 2^{-9}\\
&\gamma(2) - \gamma(6) = 2^{-1} + 2^{-5} + 2^{-7} + 2^{-9} + 2^{-13}\\
&\gamma(4)/2 = 2^{-2} + 2^{-3} - 2^{-6} - 2^{-8} - 2^{-9}
\end{aligned} \qquad (17)$$

Besides those approximations, we still have to get an integer approximated version of matrix $S\hat{W}Q$. Our investigation consist of progressively decreasing the precision of the SDCT and matrix $S\hat{W}Q$ as shown in Table II, and check the above mentioned four conditions.

Condition 1 and 4 were verified for all of the DCT versions shown in Table II. Normal entries verify all conditions whereas italic entries do not verify condition 2. The bold italic numbers verify condition 3.

TABLE II
MSE OF EACH DCT VERSION FOR INPUT GENERATED RANDOMLY

	Approximation precision of the SDCT				
Weight precision	2^{-14}	2^{-10}	2^{-9}	2^{-8}	2^{-7}
2^{-14}	0.015146	0.016952	0.038181	***0.143694***	***0.315967***
2^{-11}	0.052913	0.053977	0.073388	***0.174044***	*0.340336*
2^{-10}	0.094469	0.095234	0.112853	***0.210169***	*0.366114*
2^{-9}	*0.369178*	*0.369178*	*0.384484*	*0.471434*	*0.619028*
2^{-8}	*0.574480*	*0.574870*	*0.588552*	*0.672562*	*0.82617*

We are now interested in analyzing the PSNR degradation inserted by our integrated QWDCT solution, varying the precisions according to Table II and the quantization matrices. Thus, simulations were carried out with several video sequences to compare image fidelity.

Simulation consists of encoding video sequences using our QWDCT solution. In all integer versions of the QWDCT, depicted in Table II, we applied the 12 different quantization matrices defined in DV standard and a double precision QWDCT was implemented to serve as reference. At the decoder side, inverse quantization and double precision IDCT are applied to the coded stream. Fig. 4 shows, on average, the PSNR loss between each particular integer QWDCT and floating reference QWDCT. Simulations were accomplished for five video sequences, calendar, rugby, French fries, tree, and canoe.

As we can observe from Fig. 4, the PSNR loss is less than 1dB, and for higher quantization matrices, the quantization error is much larger than the introduced by our proposed WDCT, i.e. the quantizer acts like a noise amplifier. Another observation is that decreasing the precision of the approximations in the SW stage affects more the PSNR degradation than decreasing the precision of the SDCT. Table III shows the computational complexity of each DCT implementation.

TABLE III
OPERATIONS NUMBER TO PERFORM WQDCT

	DCT precision				
Weight precision	2^{-14}	2^{-10}	2^{-9}	2^{-8}	2^{-7}
2^{-14}	1529	1437	1429	***1361***	***1301***
2^{-11}	1421	1329	1321	***1253***	*1193*
2^{-10}	1387	1295	1287	***1219***	*1159*
2^{-9}	*1335*	*1243*	*1235*	*1167*	*1107*
2^{-8}	*1315*	*1223*	*1215*	*1147*	*1087*

In order to choose the better implementation, the target application must be considered, since quality and computational complexity varies in opposite directions, i.e. higher quality demands an higher computational complexity and vice versa. The above results show a tradeoff between requirements of video quality and computational complexity. However, regardless of DV conditions, the combination of precision of 2^{-8} and 2^{-10} for the DCT and weighting precision operations, respectively, yields an unnoticeable PSNR degradation with the lowest computational complexity. Therefore, we suggest relaxing condition 2 to a maximum tolerance value of 0.211.

V. CONCLUSION

This paper presents an integer and integrated solution for the DCT, Weighting and Quantization operations in the context of the DV coding standard. Our solution reduces the total computational complexity of those three blocks by 73%. The reduction is achieved through the use of a fast DCT implementation and by including quantization in the weighting stage. The precision achieved by our solution, compliant with the DV standard, is 2^{-9} and 2^{-10} for the SDCT and SWQ blocks, respectively. The proposed solution allows achieving a DV algorithm implementation on low power integer processors, integrated circuits, as well as on hardware reconfigurable technologies. Furthermore, our simulation results, using 720x576@25Hz video sequences, indicate that one precision condition defined in DV standard may be a bit rigid. If this condition is slightly relaxed, a small computational complexity reduction is achieved without significant loss of objective and subjective performances.

APPENDIX

The choice of QWDCT version depends of the application. However, the lowest computational complexity solution to perform our proposed QWDCT has a precision of 2^{-9} for the SDCT and 2^{-10} for the SW matrix, since this version verifies the conditions imposed by the DV standard. Thus, in order to perform (13), the approximations of elements of matrix $M_8 \otimes M_8$ are given by,

$$\begin{aligned}
\gamma(2) &= \cos(4\pi/32) = 1 - 2^{-4} - 2^{-6} + 2^{-9} \\
\gamma(4) &= \cos(8\pi/32) = 1 - 2^{-2} - 2^{-5} - 2^{-7} - 2^{-8} \\
\gamma(6) &= \cos(12\pi/32) = 2^{-2} + 2^{-3} + 2^{-7} \\
\gamma(4)\gamma(2) &= 2^{-1} + 2^{-3} + 2^{-5} - 2^{-9} \\
\gamma(6) + \gamma(2) &= 1 + 2^{-2} + 2^{-4} - 2^{-8} - 2^{-9} \\
\gamma(2) - \gamma(6) &= 2^{-1} + 2^{-5} + 2^{-7} + 2^{-9} \\
\gamma(4)/2 &= 2^{-2} + 2^{-3} - 2^{-6} - 2^{-8} - 2^{-9}
\end{aligned} \tag{18}$$

and the matrix $S\hat{W}Q$ is a symmetric matrix with elements given by,

$$\begin{aligned}
\mathrm{swq}_{00} &= 2^{-(5+q)} \\
\mathrm{swq}_{01} &= 2^{-(4+q)} + 2^{-(6+q)} \\
\mathrm{swq}_{02} &= 2^{-(3+q)} - 2^{-(6+q)} - 2^{-(9+q)} \\
\mathrm{swq}_{03} &= 2^{-(5+q)} + 2^{-(7+q)} + 2^{-(10+q)} \\
\mathrm{swq}_{04} &= 2^{-(4+q)} - 2^{-(7+q)} \\
\mathrm{swq}_{05} &= 2^{-(2+q)} - 2^{-(4+q)} + 2^{-(8+q)} + 2^{-(10+q)} \\
\mathrm{swq}_{06} &= 2^{-(5+q)} + 2^{-(8+q)} + 2^{-(10+q)} \\
\mathrm{swq}_{07} &= 2^{-(5+q)} + 2^{-(7+q)} - 2^{-(10+q)} \\
\mathrm{swq}_{11} &= 2^{-(4+q)} + 2^{-(5+q)} + 2^{-(8+q)} \\
\mathrm{swq}_{12} &= 2^{-(3+q)} + 2^{-(7+q)} \\
\mathrm{swq}_{13} &= 2^{-(4+q)} - 2^{-(7+q)} - 2^{-(8+q)} \\
\mathrm{swq}_{14} &= 2^{-(4+q)} + 2^{-(7+q)} - 2^{-(9+q)} \\
\mathrm{swq}_{15} &= 2^{-(2+q)} - 2^{-(7+q)} - 2^{-(9+q)} \\
\mathrm{swq}_{16} &= 2^{-(5+q)} + 2^{-(6+q)} - 2^{-(10+q)} \\
\mathrm{swq}_{17} &= 2^{-(5+q)} + 2^{-(6+q)} + 2^{-(10+q)} \\
\mathrm{swq}_{22} &= 2^{-(3+q)} + 2^{-(4+q)} - 2^{-(8+q)} - 2^{-(9+q)} \\
\mathrm{swq}_{23} &= 2^{-(4+q)} + 2^{-(7+q)} - 2^{-(10+q)} \\
\mathrm{swq}_{24} &= 2^{-(4+q)} + 2^{-(5+q)} \\
\mathrm{swq}_{25} &= 2^{-(2+q)} + 2^{-(4+q)} + 2^{-(6+q)} \\
\mathrm{swq}_{26} &= 2^{-(4+q)} \\
\mathrm{swq}_{27} &= 2^{-(4+q)} + 2^{-(8+q)} - 2^{-(10+q)} \\
\mathrm{swq}_{33} &= 2^{-(5+q)} - 2^{-(8+q)} - 2^{-(10+q)} \\
\mathrm{swq}_{34} &= 2^{-(5+q)} + 2^{-(8+q)} \\
\mathrm{swq}_{35} &= 2^{-(3+q)}
\end{aligned}$$

$$swq_{36} = 2^{-(5+q)} - 2^{-(7+q)}$$

$$swq_{37} = 2^{-(5+q)} - 2^{-(7+q)} + 2^{-(10+q)}$$

$$swq_{44} = 2^{-(5+q)} + 2^{-(6+q)} + 2^{-(10+q)}$$

$$swq_{45} = 2^{-(3+q)} + 2^{-(5+q)} + 2^{-(7+q)} + 2^{-(8+q)}$$

$$swq_{46} = 2^{-(5+q)} - 2^{-(10+q)}$$

$$swq_{47} = 2^{-(5+q)} + 2^{-(9+q)}$$

$$swq_{55} = 2^{-(1+q)} + 2^{-(4+q)} + 2^{-(5+q)}$$

$$swq_{56} = 2^{-(3+q)} - 2^{-(7+q)} - 2^{-(8+q)}$$

$$swq_{57} = 2^{-(3+q)} - 2^{-(7+q)} + 2^{-(10+q)}$$

$$swq_{66} = 2^{-(5+q)} - 2^{-(7+q)} - 2^{-(9+q)}$$

$$swq_{67} = 2^{-(5+q)} - 2^{-(7+q)} - 2^{-(10+q)}$$

$$swq_{77} = 2^{-(5+q)} - 2^{-(7+q)}$$

where $q = \log_2(Q)$, where Q is the quantization step obtained from Table I under each block area.

REFERENCES

[1] IEC 61384. Part 2: “SD format for 525-60 and 625-50 systems,” 1998.

[2] P. H. N de With and A. M. A. Rijckaert, “Design considerations of the video compression system of the new DV camcorder standard,” *IEEE Trans. Consumer Electron*., vol. 43, no. 4, pp 1160 – 1179, Nov. 1997.

[3] “DVCAM format overview,” Sony, http://www.sony.ca/dvcam/brochures.htm.

[4] M. A. Haque, “A two dimensional fast cosine transform,” *IEEE Trans. Acoust., Speech, and Signal Processing*, vol. 33, no. 6, pp. 1532-1539, Dec. 1985.

[5] S. C. Chan and K. L. Ho, “A new two-dimensional fast cosine transform algorithm,” *IEEE Trans. Signal Processing*, vol.39, no. 2, pp. 481-485, Feb. 1991.

[6] B. G. Lee, “A new algorithm to compute the discrete cosine transform”, *IEEE Trans. Acoust. Speech, Signal Processing*, vol. 32, pp 1243-1245, Dec. 1984.

[7] H. S. Hou, “A fast recursive algorithm for computing the discrete cosine transform,” *IEEE Trans. Acoust. Speech. Signal Processing*, vol.. 35, pp. 1455 -1461, Oct. 1987.

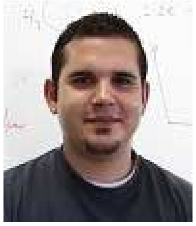
[8] P. Duhamel and C. Guillemot, “Polynomial transform computation of 2-D DCT,” *Proc. ICASSP’90*, pp. 1515 – 1518, 1990.

[9] E. Feig and S. Winograd, “Fast algorithms for the discrete cosine transform,” *IEEE Trans. Signal Processing*, vol. 40, pp 2174-2193. Sept. 1992

[10] M. Vetterli, “Fast 2-D discrete cosine transform,” *Proc. IEEE ICASSP’85*, pp. 1538–1541, 1985.

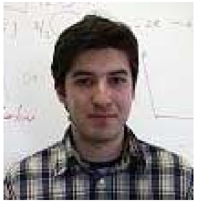
[11] V. S. Dimitrov. G.A. Jullien and W.C. Miller, “A new DCT algorithm based on encoding algebraic integers,” *ICASSP’98,* pp.1377-1380, 1998.

[12] Recommendation ITU-R BT.601-5, “Studio encoding parameters of digital television for standard 4:3 and wide-screen 19:9 aspect ratio,” 1995.



Antonio Silva graduated in Electronics and Telecommunications Engineering at Aveiro University, Portugal, in 2003.

He is now a research Assistant at the Telecommunications Institute, Portugal. His major interests are in video coding, integration, interoperability and transmission. Particularly, he has been researching fast and complexity efficient solutions.



Paulo Gouveia graduated in Electronics and Telecommunications Engineering at Aveiro University, Portugal, in 2003.

He is now a research Assistant at the Telecommunications Institute, Portugal. His major interests are in digital television, video coding, transcoding, scalability, interoperability and transmission. He has recently been involved in several international projects.

Antonio Navarro (S’89-M’97) graduated in electrical engineering from Coimbra University, Portugal in 1989. He received the MSc and PhD degrees from the University of Coimbra, Portugal and the University of Newcastle, UK in 1993 and 1996, respectively. He is currently Professor at the Electronics and Telecommunications Engineering Department of Aveiro University, Portugal. His research interests are on compression and wireless transmission of multimedia video based services. He is the Head of the Digital Television and Mobile Video research group at the Telecommunications Institute, Portugal. Dr. Navarro has participated in several national and European projects and co-authored over 50 papers. He has actively contributed to the development of several hardware/software prototypes. Dr Navarro has been participating in DVB and MPEG meetings. He has served as a consultant to the Portuguese Frequency Regulatory Body in activities of digital terrestrial TV.