A NEW DCT ALGORITHM BASED ON ENCODING ALGEBRAIC INTEGERS

V.S. Dimitrov, G.A. Jullien and W.C. Miller

VLSI Research Group
University of Windsor, Windsor, ON, Canada N9B 3P4
jullien@uwindsor.ca

ABSTRACT

In this paper we introduce an algebraic integer encoding scheme for the basis matrix elements of 8×8 DCTs and IDCTs. In particular, we encode the function $\cos\left(\frac{\pi}{16}\right)$ and generate the other matrix elements using standard trigonometric identities. This encoding technique eliminates the requirement to approximate the matrix elements; rather we use algebraic 'placeholders' for them. Using this encoding scheme we are able to produce a multiplication free implementation of the Feig-Winograd algorithm.

1. INTRODUCTION

The discrete cosine transform (DCT) plays a crucial role in several image compression standards, JPEG, ISO MPEG-1 and MPEG-2, ITU-T H.261 [1-6]. These standards make use of an 8x8 two-dimensional (2-D) DCT. Much of the research and development activities have been oriented towards efficient software implementations of algorithms for 8x8 DCT. However, for higher 'levels' of the MPEG-2 standard, (e.g. HDTV which requires 62,668,800 pixels per second [6]) VLSI hardware is the only realistic option.

For completeness, let us start with some definitions. For the 1-D DCT we adopt the definition of eqn. (1):

$$F(k) = 2\sum_{n=0}^{N-1} x(n)\cos\left[\frac{\pi(2n+1)k}{2N}\right];\ 0 \le k \le N-1 \qquad (1)$$

where $x(n)$ is a real data sequence of length N. The inverse DCT (IDCT) is defined as given in eqn. (2):

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} \bar{F}(k)\cos\left[\frac{\pi(2n+1)k}{2N}\right];\ 0 \le n \le N-1 \qquad (2)$$

where:

$$\bar{F}(k) = \begin{cases} \frac{F(0)}{2} & k=0 \\ F(k) & \text{otherwise} \end{cases}$$

The two-dimensional DCT, used extensively in image processing, can be viewed as a simple extension of the one-dimensional case, as shown in eqn. (3) (forward) and eqn. (4) (inverse.)

$$F(k,l) = \sum_{n=0}^{N-1}\sum_{m=0}^{M-1} x(n,m)\cos\left[\frac{\pi(2n+1)k}{2N}\right]\cos\left[\frac{\pi(2m+1)l}{2N}\right] \qquad (3)$$

$0 \le k \le N-1\ ;\ 0 \le l \le M-1$

$$x(n,m) = \frac{1}{NM}\sum_{k=0}^{N-1}\sum_{l=0}^{M-1} \bar{F}(k,l)\cos\left[\frac{\pi(2n+1)k}{2N}\right]\cos\left[\frac{\pi(2m+1)l}{2N}\right] \qquad (4)$$

$0 \le n \le N-1\ ;\ 0 \le m \le M-1$

where:

$$\bar{F}(k,l) = \begin{cases} \frac{F(0,0)}{4} & k=0, l=0 \\ \frac{F(k,0)}{2} & k>0,\ l=0 \\ \frac{F(0,l)}{2} & k=0, l>0 \\ F(k,l) & k>0, l>0 \end{cases}$$

and $x(n,m)$ is an $N \times M$ array of real data.

2. ALGEBRAIC-INTEGER INTERPRETATION

In [7] Cozzens and Finkelstein proposed a parallel algorithm for computation of the DFT via computation of 1) algebraic-integer quantization of the input signal (an outer level of parallelism) and 2) a residue number system (RNS) implemen-

tation of the arithmetic operations over algebraic-integers (an inner level of parallelism). Because the structure of the algebraic-integer quantization causes severe limitations over the RNS moduli, the main problem is in finding a sufficiently good compromise between these conditions. Bequilard and O'Neil used a similar idea in deriving an efficient RNS implementation of 2-D DCT [8]. Their approach is based on special method for manipulating real numbers of the form $a + b\sqrt{2}$, a, b - integers. Since the cosines of the angles, participating in 8x8 DCT can not be exactly represented in this form, the method suffers from rounding-off errors.

The main purpose of our paper is to show that the use of algebraic integers leads to fast and error-free implementation of DCT. The algorithm proposed does not require multiplications except in the final conversion from algebraic-integer to real form.

A direct implementation of the 2-D DCT requires N^4 multiplications. By noting that each cosine part only varies with one of the summations, the transform can be executed by a row-column decomposition with only $2N^3$ multiplications, that is, $2N$ multiplications per pixel.

The elements of the transformation matrix are real numbers of the form $\cos\left(\frac{\text{integer} \cdot \pi}{16}\right)$. Rather than using approximations to these elements (the classical procedure), we adopt another encoding scheme oriented towards processing numbers of this particular form. The approach can be straightforwardly extended to other values of N.

Let us look at the 'first' nonzero angle, that is $\frac{\pi}{16}$. We can rewrite $\cos\left(\frac{\pi}{16}\right)$ in the form:

$$\cos\left(\frac{\pi}{16}\right) = \frac{\sqrt{2 + \sqrt{2 + \sqrt{2}}}}{2} \tag{5}$$

The remainder of the cosines participating in a 1-D 8-point DCT definition can be similarly represented:

$$\cos\left(\frac{2 \cdot \pi}{16}\right) = \frac{\sqrt{2 + \sqrt{2}}}{2} \tag{6}$$

$$\cos\left(\frac{3 \cdot \pi}{16}\right) = \frac{\sqrt{2 + \sqrt{2 - \sqrt{2}}}}{2} \tag{7}$$

$$\cos\left(\frac{4 \cdot \pi}{16}\right) = \frac{\sqrt{2}}{2} \tag{8}$$

$$\cos\left(\frac{5 \cdot \pi}{16}\right) = \frac{\sqrt{2 - \sqrt{2 - \sqrt{2}}}}{2} \tag{9}$$

$$\cos\left(\frac{6 \cdot \pi}{16}\right) = \frac{\sqrt{2 - \sqrt{2}}}{2} \tag{10}$$

$$\cos\left(\frac{7 \cdot \pi}{16}\right) = \frac{\sqrt{2 - \sqrt{2 + \sqrt{2}}}}{2} \tag{11}$$

The other cosines can be easily computed by taking into account the corresponding symmetries.

First of all, without compromising the calculations, we omit the '2' in the denominators. Let us denote $z = 2 \cdot \cos\left(\frac{\pi}{16}\right) = \sqrt{2 + \sqrt{2 + \sqrt{2}}}$. z is a root of eqn. (12):

$$x^8 - 8x^4 + 20x^2 - 16x^2 + 2 = 0 \tag{12}$$

The main trick is that the other numbers we would like to handle (that is, $2 \cdot \cos\left(\frac{\text{integer} \cdot \pi}{16}\right)$ can be represented as a polynomial of z.

Let us consider the following polynomial:

$$f(z) = \sum_{i=0}^{7} a_i z^i \tag{13}$$

where a_i are integers. Definitely, z, that is $\sqrt{2 + \sqrt{2 + \sqrt{2}}}$ corresponds to the following particular choice for a_i: $(0, 1, 0, 0, 0, 0, 0, 0)$. Therefore, we have an *exact* code for z. For the other cosines, it turns out that we can represent them *exactly* as a combinations of eight small integers a_i. Table 1 provides the corresponding coefficients:.

The other cosines can be obtained by changing the signs of the numbers given, say, $\cos\left(\frac{9 \cdot \pi}{16}\right) = -\cos\left(\frac{7 \cdot \pi}{16}\right)$, therefore, $\cos\left(\frac{9 \cdot \pi}{16}\right) \rightarrow (0, 7, 0, -14, 0, 7, 0, -1)$.

We have therefore produced an error-free (infinite precision) representation of the coefficients necessary to evalu-

ate 1-D and 2-D DCTs.

Table 1: The exact representation of the cosines participating in the 8-point DCT

$2\cos\frac{\text{integer}\cdot\pi}{16}$	a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7
$2\cdot\cos\frac{1\cdot\pi}{16}$	0	1	0	0	0	0	0	0
$2\cdot\cos\frac{2\cdot\pi}{16}$	-2	0	1	0	0	0	0	0
$2\cdot\cos\frac{3\cdot\pi}{16}$	0	-3	0	1	0	0	0	0
$2\cdot\cos\frac{4\cdot\pi}{16}$	2	0	-4	0	1	0	0	0
$2\cdot\cos\frac{5\cdot\pi}{16}$	0	5	0	-5	0	1	0	0
$2\cdot\cos\frac{6\cdot\pi}{16}$	-2	0	9	0	-6	0	1	0
$2\cdot\cos\frac{7\cdot\pi}{16}$	0	-7	0	14	0	-7	0	1

The next section discusses arithmetic operations with this number representation.

3. ARITHMETIC OPERATIONS

The real numbers of the form of eqn. (5) form a ring, which we shall denote as $Z\left[\sqrt{2+\sqrt{2+\sqrt{2}}}\right]$. The addition in this ring is componentwise. The multiplication is equivalent to a polynomial multiplication modulo $z^8 - 8z^4 + 20z^2 - 16z^2 + 2$.

Now let us return to the original problem. The input data to the DCT or IDCT are codes of the corresponding pixels, non-negative integers, and the multiplications we require are very simple. For example, if we have to multiply s times $2\cdot\cos\left(\frac{6\cdot\pi}{16}\right)$, this is equivalent to the operation shown in Table 2.

It is easy to see that the multiplications by the integers in Table 1 can be performed using, in the worst case, 1 addition/subtraction and 2 shifts. This worst case appears when one needs to multiply s times 14 or 6.

For the inverse map, given a set of eight integers $(a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7)$, we require to find the value of the polynomial $f(z) = \sum_{i=0}^{7} a_i z^i$ for $z = \sqrt{2+\sqrt{2+\sqrt{2}}}$.

Table 2: Multiplication of integer number s times $2\cdot\cos\left(\frac{6\cdot\pi}{16}\right)$, performed in $Z\left[\sqrt{2+\sqrt{2+\sqrt{2}}}\right]$

s	$2\cdot\cos\left(\frac{6\cdot\pi}{16}\right)$	$2\cdot s\cdot\cos\left(\frac{6\cdot\pi}{16}\right)$
$\begin{pmatrix} s, 0, 0, 0 \\ 0, 0, 0, 0 \end{pmatrix}$	$\begin{pmatrix} -2, 0, 9, 0 \\ -6, 0, 1, 0 \end{pmatrix}$	$\begin{pmatrix} -2s, 0, 9s, 0 \\ -6s, 0, s, 0 \end{pmatrix}$

For the computation of a 1-D or 2-D DCT (or IDCT), what we actually need is to recover the integer part of the result and the most significant bit of the fractional part, to allow correct rounding. In this case we may use the fact, that z can be approximated with very good precision in the form:

$$z \approx 2 - 2^{-5} - 2^{-7} + 2^{-11} \tag{14}$$

Summarizing, we have the following:

1. We may compute a 1-D 8-point DCT very quickly.
2. For 2-D 8×8 DCTs we have several alternatives, and we identify 4 possible alternative 2-D procedures below:

   a) Eight 8-point DCTs (row-wise) - very fast; re-code (polynomial evaluation); eight 8-point DCTs (column-wise) - very fast; re-code (polynomial evaluation.)

   b) Eight 8-point DCTs (row-wise) - very fast; eight 8-point DCTs (column-wise) requiring polynomial multiplication where one of the polynomials is fixed; re-code (polynomial evaluation.)

   c) Compute 8x8 DCT as a sixty four 64-point inner products, using the fact that the products of the cosines can be expressed efficiently as octaves.

   d) Direct approach in computing 8x8 DCT

4. FEIG-WINOGRAD 8X8 DCT ALGORITHM

Given our present knowledge, a direct approach for computing 8×8 DCTs and IDCTs appears the most promising. There exist a variety of techniques, allowing us to calculate 2-D DCTs and IDCTs avoiding row-column techniques. In our opinion, the most promising algorithm is that proposed by Feig and Winograd [10]. We do not explain it here, the reader may find the details in [10]. We will only briefly mention the general structure of the algorithm and the application of our algebraic-integer encoding scheme.

1. Step 1 - preadditions stage, with 8-bit integers.
2. Step 2 - multiplication stage. This step requires multipli-

cations by $\cos\left[\frac{\pi}{8}\right]$, $\cos\left[\frac{\pi}{4}\right]$ and $\cos\left[\frac{3\pi}{8}\right]$. Using the techniques in section 3, these can be performed in an error-free manner. The Feig-Winograd algorithm does not use angles that are odd multiples of $\pi/16$. Therefore, the cosines we need can be simplified in an algebraic-integer form as shown in Table 3:

Table 3: The algebraic-integer representation of cosines for Feig-Winograd 2-D DCT and IDCT algorithms

Element	a_0	a_1	a_2	a_3
$2\cos\pi/8$	0	1	0	0
$2\cos 2\pi/8$	-2	0	1	0
$2\cos 3\pi/8$	0	-3	0	1

The advantages of the Feig-Winograd algorithm are:

a) We need only quadruples (not octaves) to represent the cosines used in the computations.

b) The maximal number we have to multiply by is 3.

c) The final reconstruction uses a polynomial of third (not seventh) degree.

d) The representation of $2\cos\frac{1*\pi}{8} = \sqrt{2+\sqrt{2}}$ in canonic-signed digit form needs fewer ones (assuming 12-bit precision) than $2\cos\frac{1*\pi}{16} = \sqrt{2+\sqrt{2+\sqrt{2}}}$.

3. Step 3 - postadditions stages. This requires the additions of quadruples, which is implementing in a component-wise manner.

The final reconstruction depends on the precision used to represent $\bar{z} = \sqrt{2+\sqrt{2}}$. Since the final result is in an error free format, we can easily estimate the precision we need to guarantee sufficient accuracy. If the input and output data are 8-bits maximum, then the representation of $\bar{z}$ as $\bar{z} = 10.00\bar{1}0\bar{1}0010000\ldots \approx 2 - 2^{-3} - {}^{-5} + 2^{-9}$ is sufficient.

Table 4 shows estimations for our algorithm and comparisons with the arithmetic complexity of Feig-Winograd [10] and Duhamel-Guillemot [10] algorithms.

5. CONCLUSIONS

In this paper we have proposed a new approach aimed at efficient computation of 8×8 DCTs and IDCTs. The approach is based on encoding the basis set using algebraic integers. The main advantages of the method proposed are: 1) it is error-free up until the final reconstruction; 2) it needs no multiplication, which is very suitable from VLSI view point; 3) it can be combined with many already existing algorithms for DCT and IDCT. We have found the most suitable 2-D DCT algorithm for this encoding scheme to be the Feig-Winograd algorithm.

Table 4: Comparisons for computing 8x8 DCTs

Algorithm	Multiplications	Additions
Feig-Winograd	54 (16 bit)	462 (16 bit)
Duhamel-Guillemot	96 (16 bit)	484 (16 bit)
Algebraic-integer	None	636 (12 bit)

6. REFERENCES

[1] S.Winograd, "Arithmetic Complexity of Computations", CBMS-NSF Conf. Ser. in Appl. Math., 1980.

[2] J.H.Cozzens and L.A.Finkelstein, "Computing the discrete Fourier transform using residue number systems in a ring of algebraic integers", IEEE Trans. on Inf. Th., 31, pp.580-588, 1985

[3] N.Ahmed, T.Natarajan and K.R.Rao, "Discrete cosine transform", IEEE Trans. Comput. C-23, pp.90-93,1974

[4] K.R.Rao and P.Yip, "Discrete Cosine Transform - Algorithms, Advantages, Applications", New York, Academic - 1990

[5] E.Feig and S.Winograd, "Fast algorithms for the discrete cosine transform", IEEE Trans. Sig. Proc., vol.40, No.9, pp.2174-2193, Sep. 1992.

[6] "CCITT Recommendation H.261", 1990

[7] D.L.Gall, "MPEG: a video compression standard for multimedia applications", Comm. ACM, 34, pp.46-58,1991

[8] P.Duhamel and C.Guillemot, "Polynomial transform computation for the 2-D DCT", Proc. ICASSP-90 (Albuquerque, NM), pp.1515-1518, 1990.

[9] A.L.Bequilard and S.D.O'Neil, "Systolic RNS computation of the two-dimensional DCT in a ring of algebraic integers", Proc. of the 20th Annual Conference on Inf. Sc. and Syst., pp.783-789, 1986.

[10] G.K.Wallace, "The JPEG still picture compression standard", Comm. ACM, vol.34, pp.31-44, 1991.

[11] G.Taylor and G.M.Blair, "VLSI module design for the discrete cosine transform", to be published in IEE, Proc. Computers and Digital Techniques