LOW COMPLEXITY FIXED-POINT APPROXIMATION OF INVERSE DISCRETE COSINE TRANSFORM

Yuriy A. Reznik, Sr. Member IEEE, Arianne T. Hinds[§], Sr. Member IEEE, Nenad Rijavec[§]*

*QUALCOMM, Incorporated, San Diego, CA, Email: yreznik@qualcomm.com
[§]IBM, Incorporated, Boulder, CO, Email: arianne,nenad@us.ibm.com

ABSTRACT

This paper presents an efficient algorithm for computing the Inverse Discrete Cosine Transform (IDCT) for image and video coding applications. This algorithm was submitted in response to MPEG's call for proposals for ISO/IEC 23002-2 (Fixed-Point 8x8 IDCT and DCT) standard, and was subsequently adopted in the Working Draft 1 of this standard. Our proposed algorithm is a multiplication-free implementation. It is based on a modification of Arai, Agui, and Nakajima's (AAN) factorization, and requires only 42 addition and 16 shift operations per scaled 1D transform. Each register in our scaled 1D transform requires at most 22 bits. This implementation complies with the MPEG IDCT precision specification ISO/IEC 23002-1.

Index Terms — DCT, IDCT, factorization, multiplier-less algorithms

1. INTRODUCTION

The Moving Picture Experts Group (MPEG) is currently developing a standard for a fixed-point approximation of 8x8 Inverse Discrete Cosine Transform (IDCT). The use of this IDCT specification in implementing existing MPEG video coding standards (MPEG-1, MPEG-2, and MPEG-4 part 2) should be voluntary and intends to offer the following benefits [1]:

- Providing an example IDCT method to ease the implementation community in their design of decoders and encoders.
- To help ensure that decoders are implemented in conformance with the standard, as those decoders that are designed to use the specified method will be assured to conform to the IDCT conformance requirements of the relevant video coding standards.
- To improve the quality of delivered video, as encoders designed to target their encoding process for the specified IDCT method can be assured that the decoding process will be free of *drift* on all decoders that conform to the new standard.

The call for proposals (CfP) issued by MPEG [1] requires that any candidate for the new standard must meet or exceed certain accuracy thresholds relative to the integer-valued IDCT, defined as follows:

$$f[y][x] = \left\lfloor \sum_{u=0}^{7}\sum_{v=0}^{7} F[v][u]\,\frac{C_u C_v}{4}\cos\left(\frac{(2x+1)u\pi}{16}\right)\cos\left(\frac{(2y+1)v\pi}{16}\right) + \frac{1}{2} \right\rfloor, \quad (1)$$

where:

- $c_u = 1/\sqrt{2}$ for $u = 0$, otherwise 1,
- $c_v = 1/\sqrt{2}$ for $v = 0$, otherwise 1,
- *F[v][u]* are input DCT coefficients with values in the range of [-2048, 2047].
- *f*[*y*][x] are reconstructed pixel values.

The IDCT accuracy thresholds are defined for the following metrics [1,2]:

- p – maximum absolute difference between reconstructed pixels (specification requires $p \le 1$);
- *d*[*x,y*] – average differences between pixels (specification requires $|d[x,y]| \le 0.015$ for all [x,y]);
- m – average differences between pixels (specification requires: $|m| \le 0.0015$);
- e[x,y] – average square difference between pixels (specification requires $|e[x,y]| \le 0.06$ for all [x,y]);
- n – average of all square differences between pixels (specification requires: $|n| \le 0.02$).

The CfP specifies that each of these metrics shall be collected over randomly generated blocks of input. These blocks are generated using a random block generator defined in [2]. Results are collected over sets of 10,000 blocks and 1,000,000 blocks, where each set is generated once for each of five different input ranges.

Beyond the above specifications, the call for proposals [1] lists additional evaluation criteria, including resources required for implementation on hardware and software platforms, simplicity, etc.

The remainder of this paper will review our IDCT design, submitted in response to call for proposals [1], and

selected for the Working Draft 1 of this new standard [3]. Section 2 of this paper describes the architecture and underlying factorization selected for our fixed-point design. Section 3 elaborates on the methodology used to complete the design. Section 4 provides the performance results for the above testing process. Section 5 presents our summary.

2. ARCHITECTURE AND FACTORIZATION

In our IDCT, we adopt a *scaled, separable* architecture. The key benefits of such an architecture include:

- Lower multiplicative complexity due to merged multiplications in a separate scaling phase.
- Possible further reduction in complexity due to the ability to merge the scaling operations in with quantization processes in implementations of JPEG, H.263 and MPEG-4 (Part 2) standards.
- High precision due to the ability to minimize and distribute errors of fixed-point approximations of cosine multiplications within 1D transforms by adjusting the scale factors.

We use a separate scaling step to pre-multiply all input coefficients by a certain quantity $C = 2^P$, serving as a fixed-point "mantissa" for subsequent 1D IDCT computations. To achieve even higher precision of scaling, we use S=P+R bits to convert floating point scale factors to integers, and execute shift right operations by R bits after multiplications. In our proposed implementation, parameter *S* is chosen to be 15 thereby facilitating simple implementations on platforms with signed/unsigned 16-bit multipliers. To achieve a proper rounding we add the quantity 2^{P-1} to the DC coefficient right after scaling.

After scaling, we execute 16 iterations of our scaled 1D IDCT over all columns and rows in the 8x8 matrix. This 1D IDCT is implemented using only 16 shift and 42 addition operations, and *P=10* bits of added (during the scale step) precision as the fixed-point "mantissa". Finally, after the cascade of 1D IDCTs we shift all quantities in the 8x8 matrix by P bits to the right.

The complete flow-graph of the factorization used in our algorithm is presented in Fig. 1. It can be seen that generally it is very similar to a well-known Arai, Agui, and Nakajima's (AAN)-factorization of scaled IDCT [4]. One difference in our implementation is that the 3-multiplication section in the odd part of the AAN design is replaced by a butterfly, which turns out to be beneficial for multiplier-less implementations.

The flow-graph of our IDCT factorization is shown in Fig 1.

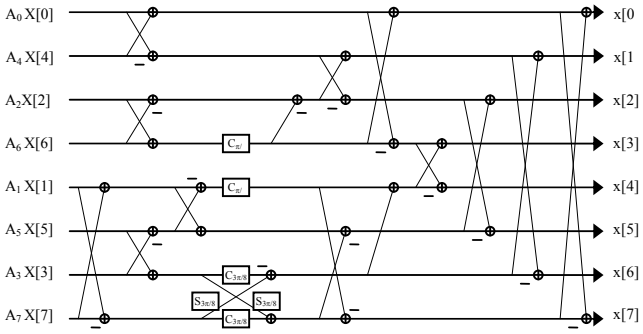

Fig 1. The flow-graph of a scaled 1D IDCT adopted in proposal.

The values of the coefficients and scale factors in this flow-graph are defined in Table 1 :

TABLE I. CONSTANTS USED IN PROPOSED IDCT

Constant	Value
$C_{\pi/4}$	$\cos(\pi/4) \approx 0.707106781$
$C_{3\pi/8}$	$\cos(3\pi/8) \approx 0.382683432$
$S_{3\pi/8}$	$\sin(3\pi/8) \approx 0.923879533$
A_0	$\frac{1}{2\sqrt{2}} \approx 0.3535533906$
A_1	$\frac{\cos(7\pi/16)}{2\sin(3\pi/8)-\sqrt{2}} \approx 0.4499881115$
A_2	$\frac{\cos(\pi/8)}{\sqrt{2}} \approx 0.6532814824$
A_3	$\frac{\cos(5\pi/16)}{\sqrt{2}+2\cos(3\pi/8)} \approx 0.2548977895$
A_4	$\frac{1}{2\sqrt{2}} \approx 0.3535533906$
A_5	$\frac{\cos(3\pi/16)}{\sqrt{2}-2\cos(3\pi/8)} \approx 1.2814577239$
A_6	$\frac{\cos(3\pi/8)}{\sqrt{2}} \approx 0.2705980501$
A_7	$\frac{\cos(\pi/16)}{\sqrt{2}+2\sin(3\pi/8)} \approx 0.3006724435$

In Fig. 2, we show a flow-graph of our fixed-point implementation. In this flow-graph boxes α_1, β_1, α_2, β_2, γ, and δ denote fixed-point approximations of multiplications by the corresponding constants in the original IDCT flow-graph (cf. Fig 1). Note that the value corresponding to both α_1 and α_2 is 1.

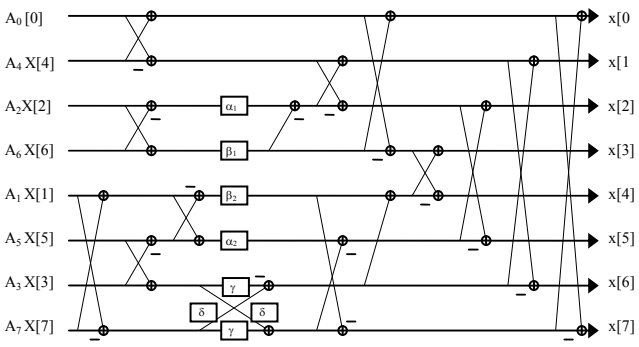

Fig 2. Flow-graph for fixed-point approximation of scaled 1D IDCT.

3. FIXED-POINT DESIGN METHODOLOGY

To achieve our fixed-point design, we first separate the irrational constants into two groups, based on the data dependencies in the flow-graph shown in Fig. 2. Group 1 consists of constants α_1 and β_1. Group 2 consists of constants $\alpha_2, \beta_2, \gamma,$ and δ. By forming these groups, we approximate each group separately thereby allowing us additional degrees of freedom in determining the error we will tolerate for each group.

Next, we employ a modified version of the bicriterial optimization methods described in [5,6] to identify rational approximations for each of the two groups. Using these techniques, we arrive at a separate scale factor to be applied to each group. As shown in Table 1 for this design of our IDCT, the scale factor is the same for each group, but in practice this is rarely the case.

Our modified bicriterial optimization approach searches a candidate space of integer approximations for each constant in each group, and factors out of each set of integers a floating point constant that can be used as a common factor for the entire group. This factor serves to scale the worst case absolute error for the entire group of approximations, so that we can weigh this error with the implementation cost computed for the group in terms of addition operations. In general, as the implementation cost of a set of candidate approximations for a group goes up, the associated worst case approximation error for that group goes down. Hence, using these techniques we are able to find the Pareto-optimal front of approximations, where each approximation has a different number of required operations and each approximation results in the minimum approximation error for the corresponding number of operations. For any approximation not in the front, there is an approximation in the front such that it requires an equal number of operations, but with approximation error that is less than or equal to the error in the non-front approximation being considered.

Moreover in computing the implementation costs for each candidate solution set (and defining final algorithms for computing the products) we assume that some intermediate values can be stored and subsequently reused. This way we arrive at the algorithms shown in Table 2. For example, the term x2 is computed once and reused to compute both the values 3135 and 473 in our design. Reuse of common expressions such as these can reduce the total complexity in terms of addition operations, especially when these expressions can be used to compute two constants simultaneously.

Next, we scale the approximations in each group and correspondingly each group's scale factor by a power of two so that our approximations are now implemented as dyadic rationals, i.e. with shift right operations instead of shift left operations. The resulting group factors are then folded directly into the original AAN scale factors shown with Fig. 1 and defined in Table 2. These combined scale factors are then converted to fixed-precision using the parameter S.

4. PERFORMANCE RESULTS AND COMPLEXITY ANALYSIS

As required in [1], we measure the performance of our IDCT approximation by computing the metrics described above. In Table III, we report the worst case results for each of the metrics measured across all tests, (with the exception of the maximum absolute difference between reconstructed pixels which is 1.0 for all tests). We report these metrics for our approximation with the fixed point scale factors computed both with S=15 and S=16. Note that the implementation chosen for the initial working daft of the standard is the algorithm with identifier "A1" As shown in the table, all results fall within the tolerances acceptable for the new standard [3].

We report the complexity of our algorithm in Table III in terms of 1D and 2D complexity. For 2D complexity, we compute the total cost in terms of addition (denoted by 'a') and shift (denoted by 's') operations incurred over the cascade of 16 iterations. To this sum, we add the complexity of our scale step. The term bit-adds denotes the value obtained by multiplying the number of addition operations required by their required width in terms of bits.

4.1. Complexity of scaling step

We use the parameter 'k' to denote the number of nonzero coefficients to which the scaling step is applied, and 'm' to denote multiplication operations. The input data to the IDCT process in video decoders originate from a list of non-zero coefficients in the 8x8 block. For convenience, we assume that there is a total of k such coefficients. In practice, this number k is typically small.

TABLE II. DETAILS OF FIXED POINT APPROXIMATIONS USED IN IDCT WITH 42 ADDITIONS AND 16 SHIFTS

C	Original Value	Rational Approx.	Group's Scale factor	Algorithms: x=x*[α,β,γ];y=x*δ	Complexity	
					Add-s	Shifts
α_1	1	1	1.0000442471	x=x;	0	0

β_1	cos(π)	181/256		x2=x+(x>>2); // 101
				x3=x-(x2>>2); // 01011
				x =x3+(x2>>6); // 010110101
					3	3

α_1	1	1	1.0000442471	x=x;	0	0

β_2	cos(π/4)	181/256		x2=x+(x>>2); // 101
				x3=x-(x2>>2); // 01011
				x =x3+(x2>>6); // 010110101
					3	3

γ	cos(3π/8)	3135/8192		x2=x-(x>>4); // 01111
				x3=x2+(x>>10); // 01111000001
				x =(x-(x3>>2))>>1; // 00110000111111
				y =x3-(x2>>6); // 0111011001
					4	5

δ	sin(3π/8)	473/512				

TABLE III. ACCURACY MEASUREMENTS AND COMPEXITY ANALYSIS

Algorithm			**Precision**					**Complexity**		
ID	S	P	max e[x,y] (0.06)	N (0.02)	max |d[x,y]| (0.015)	M (0.0015)	near DC	1D		2D (ops)
								bit-adds	Ops	
A1	15	10	0.01600	0.01030	0.00950	0.00039	0	924	42a,16s	km,673a, (320+k)s
A2	16	10	0.01380	0.00906	0.01070	0.00048	0			

Since the first step in a scaled 2D IDCT architecture is essentially a multiplication by scale-factors, these multiplications can now be performed only for k non zero coefficients. Therefore instead of executing 64 multiplications during the scaling stage, in a typical video decoding scenario it would be sufficient to execute only k multiplications which can be as small as 4 or 5.

In order to realize this savings, a decoder simply needs to pass its list of non-zero coefficients to the transform. In turn, transform scaling can be easily implemented in such a way that it places the results of multiplications in the appropriate locations in the 8x8 matrix, thereby preparing it for the remaining stages in the transform.

Since inverse quantization and scaling essentially perform successive multiplications of coefficients by known constant factors (quantization parameter, factors from weighting matrices, and IDCT scale factors), they can (in most of instances) be simply merged into a single multiplication by a pre-computed product of all involved intermediate constants.

5. SUMMARY

This paper has shown that by application of the above methodology in conjunction with a scaled architecture, we can derive a low complexity fixed-point approximation of the IDCT with accuracy sufficient to meet the requirements of the accuracy standard [2]. Our algorithm A1 was recognized as one having the lowest complexity among submitted scaled IDCT proposals, and it was selected for the Working Draft 1 of this new standard [3].

6. REFERENCES

[1] ISO/IEC JTC1/SC29/WG11 N7335, Call for Proposals on Fixed-Point IDCT and DCT Standard, Poznan, Poland, July 2005.

[2] ISO/IEC JTC1/SC29/WG11 N7815 [23002-1 FDIS] Information technology – MPEG video technologies – Part 1: Accuracy requirements for implementation of integer-output 8x8 inverse discrete cosine transform.

[3] ISO/IEC JTC1/SC29/WG11 N7817 [23002-2 WD1] Information technology – MPEG Video Technologies – Part 2: Fixed-point 8x8 IDCT and DCT transforms.

[4] Y. Arai, T. Agui, and M. Nakajima, "A Fast DCT-SQ Scheme for Images", *Transactions of the IEICE* E 71(11):1095, November 1988.

[5] N. Rijavec, A.T. Hinds, "Multicriterial Optimization Approach to Eliminating Multiplications", *Proceedings of the 2006 8th IEEE Workshop on Multimedia Signal Processing,* 2006, pp. 368-371.

[6] A.T. Hinds, and J.L. Mitchell, "A Fast and Accurate Inverse Discrete Cosine Transform", *Proceedings of the IEEE Workshop on Signal Processing Systems*, November 2005, pp. 87-93.