MULTIPLIERLESS APPROXIMATION OF FAST DCT ALGORITHMS

Raymond K.W. Chan   *Moon-Chuen Lee*

Department of Computer Science and Engineering,
The Chinese University of Hong Kong, Shatin, N.T., Hong Kong.

ABSTRACT

This paper proposes an effective method for converting any fast DCT algorithm into an approximate multiplierless version. Basically it approximates any constant in the original transform by a signed digit representation. We developed an efficient algorithm to convert any constant into a signed digit string with a minimum number of non-zero signed digits and a reduced length. As the accuracy of an approximated algorithm depends critically on the assignment of signed digits to the constants, this paper formulated an effective algorithm for finding an effective signed digits configuration which could minimize the MSE of an approximated DCT algorithm with a specified complexity. Experiment results show that the AAN's fast DCT algorithm, approximated by the proposed method and using an optimized configuration can be used to reconstruct images with high visual quality in terms of PSNR.

1. INTRODUCTION

In signal processing, fast algorithms of 1-D DCT has been investigated extensively. In image and video signal processing, 2-D DCTs are being widely used. Since a 2-D DCT is separable in its matrix form, it can be implemented by the row-column application of the 1-D DCT.

In order to speed up the transform, many fast algorithms have been proposed for floating-point DCT [1]. To achieve even faster transform, multiplierless implementation has been investigated for more than one decade. In [2], the eight-point integer DCT (ICT) used in image and video-coding standards was developed based on the principle of dyadic symmetry. However, this method can hardly be used to implement fast DCT algorithms. Tran [3] and others [4,5] proposed techniques to convert fast floating-point DCT algorithms of a certain type into their approximate integer or multiplierless versions, and developed two integer DCTs known as BinDCT and IntDCT respectively.

Chan and Yiu [6] proposed a family of multiplierless discrete cosine and sine transforms represented in sum-of-power-of-two (SOPOT) form. They approximated Wang's fast algorithm [7] by first factorizing the rotation matrices in the fast algorithm to triangular matrices and then converting them into the SOPOT representation. They used a random search to find the SOPOT coefficients.

However, the foregoing methods require the presence of rotation matrices or butterfly structures in any fast algorithm to be converted. So, they cannot be used to approximate other fast algorithms such as AAN's [8] and Lee's [9] DCT algorithms, which do not have the required structures. Moreover, the above publications only approximate an algorithm's kernel constants; it is expected they still need to use real arithmetic in the normalization step. This paper addresses two important issues: (1) how to convert fast algorithms without butterfly structures; (2) how to assign signed digits to approximate the constants in order to minimize the errors of an approximated DCT algorithm having a certain target complexity.

2. MULTIPLIERLESS DCT CONVERSION

DCT is a kind of linear transform; this section introduces a method for linear transform approximation which is related to our proposed methodology for converting any fast DCT algorithm into an approximate multiplierless version.

2.1 Linear Transform Approximation

Any real number c can be approximated by a sum of several power-of-two terms; that is, $c = \sum_{i=k}^{m} a_i 2^i + r$ where $a_i \in \{-1, 0, 1\}$, k and m are positive integers, and r is the remainder term or the error term. Any float operation $c \cdot x$ involving a constant multiplied by a data point x can be converted into a number of shift-and-add operations on x by approximating c with a number of power-of-two terms. For any linear transform $Y = C \cdot X$ involving a real constant multiplication, where X, C, and Y are the domain vector, the transform matrix consisting of real constants, and the range vector respectively, we can apply the above technique to convert each of the real constant multiplications into shift-and-add operations; then a multiplierless version of the transform can be obtained. This kind of conversion can, however, introduce an approximation error. The size of the error would depend on the magnitude of r.

To reduce the complexity of a linear transform, we need to approximate each constant with the minimum number of non-zero digits. The algorithm below converts any constant

1-4244-0367-7/06/$20.00 ©2006 IEEE

c into its signed digit representation with the minimum number of non-zero signed digits. This algorithm has a complexity *O*(*n*), where *n* is the desired number of non-zero digits.

2.2 Minimum Signed Digits Conversion Algorithm

The proposed algorithm for finding the MSD for a given constant *c* consists of the following steps:

```
1. Let $S = \{\varnothing\}$   // the set of binary tuples for non-zero signed digits.
2. if $c \geq 0$ then $m = \lfloor \log_2 c \rfloor$ else $m = -\lfloor \log_2 (-c) \rfloor$;
3. find minimum of $c' = \left| c + (s)2^{(m+i)} \right|$ where $s \in \{-1,1\}$, and $i \in \{-1,0,1\}$;
4. Add $(s, m+i)$ to $S$; let $c = c'$ ;
5. if termination condition not satisfied then goto 2;
6. Convert the tuples in $S$ into a signed digits string;
7. Replace the most significant digits in the signed digit string having the
   pattern $10\overline{1}$ by $11$
```

The above algorithm generates the minimum number of non-zero signed digits for a given real constant. It is actually a greedy algorithm that picks the most significant digit in each iteration as a signed digit. It is different from the CSD [10] generation method that eliminates consecutive non-zero digits from their binary representation. Besides, the power-of-two terms generated here are in decreasing order in magnitude. For any power-of-two term generated, it is larger than the sum of other terms generated afterwards. Mathematically, we have $2^m > \sum_{i=k}^{m-1} 2^i$, for all integer *m* and *k*<*m*. The last step of the above algorithm is an attempt to reduce the length of the signed digits representation of a constant by 1 without jeopardizing its accuracy.

Below are three possible termination conditions for the above algorithm:

1. A specified number of non-zero digits (*d*) has been found.
2. The approximation error (*r*) is less than a given value (ε).
3. A certain power-of-two index (*m*) has been reached.

For example, to approximate the number 3.141592654 with the proposed MSD representation, the respective signed digit strings based on the above three different termination conditions will be:

1. 11.0010010001 -- termination condition with $d = 5$;
2. 11.001001 -- termination condition with $\varepsilon = 10^{-3}$;
3. 11.001 -- termination condition with $m = 2^{-3}$

The last step of the above algorithm would convert each of the above signed digit strings into an equivalent one with its length reduced by 1. For instance, the signed digit string $10\overline{1}.0010010001$ would be changed to 11.0010010001. In converting a DCT algorithm into an approximate multiplierless version, we can employ the above technique to find an MSD representation for each constant. The number of digits actually used to approximate a constant would depend on the approximation error allowed. In general, if more digits are used, we have a better approximation of the constant, and more shift-and-add operations. Therefore, to determine the number of digits to be used, we need to consider the allowable approximation error, and the target complexity of the algorithm.

3. CONVERSION OF AAN'S FAST ALGORITHM

The scaled DCT fast algorithm proposed by AAN [8] has the smallest number of multiplications. Its kernel requires only 5 multiplications, and its normalization step involving eight multiplications is expected to be incorporated into the quantization step. Fig. 1 shows the original signal flow diagram of AAN's forward transform algorithm. It contains a butterfly structure involving C_2 and C_6 and two non-butterfly structures involving C_4. A butterfly structure has the form shown in Fig.2a, which is equivalent to the structure shown in Fig. 2b, which can be further converted into the lifting steps as shown in Fig. 2c. We converted all AAN's butterfly structures into lifting steps before using our proposed method to approximate the DCT algorithm.

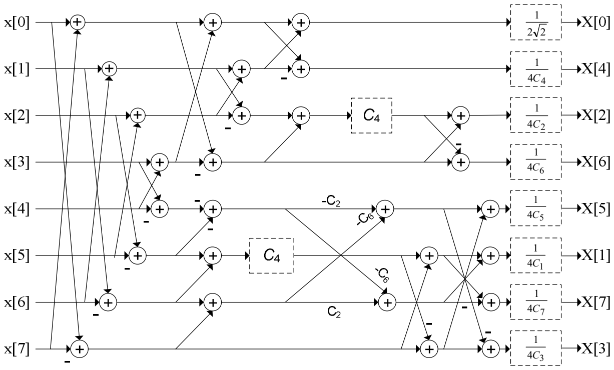

Fig. 1. Forward AAN's fast algorithm

Tran [3] proposed a method to convert Chen & Smith's [11] and LLM's [12] DCT algorithms, with each multiplication inside a butterfly structure, into their approximate multiplierless versions. However, this method cannot convert any DCT algorithm having a constant not included in a butterfly structure. In AAN's DCT algorithm, there are two non-bufferfly structures involving a multiplication with the constant C_4 as shown in Fig.1. These non-bufferfly structures violate the conditions $r_{11} \neq 0$ or $r_{11}r_{22} - r_{21}r_{22} \neq 0$ in [3] required for using Tran's conversion method which, therefore, cannot be employed to approximate AAN's algorithms.

However, using our proposed method, we can replace the multiplications involving the constants $C_4, \rho_1, \mu_1, \rho_2$ by shift-and-add operations only. Then we can make AAN's DCT kernel free from real arithmetic. Thus, the approximate multiplierless version of the forward transform and that of the inverse transform can be easily obtained. We can also convert all the multiplications involving the normalization constants into shift-and-add operations so that the whole fast algorithm can be made multiplierless.

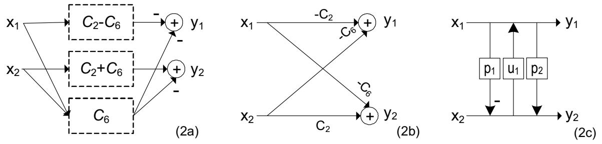

Fig. 2. Converting butterfly structure (2b) obtained from (2a), into lifting step structure (2c).

4. FINDING A SIGNED DIGITS CONFIGURATION

Suppose there are n constants C_1, C_2, …and C_n in a fast DCT algorithm and they are assigned d_1, d_2, …and d_n signed digits respectively. Then (d_1, d_2, ...d_n) is called a *signed digits configuration* of the algorithm being approximated. The total number of all the signed digits is defined as the *length* of the configuration. The complexity of the algorithm depends partly on the *length* of its signed digits configuration. An algorithm approximated by two different configurations should have the same complexity as long as they have the same length. However, the two different configurations with the same length may lead to different MSEs. The reason is that the MSE of an algorithm can be more sensitive to some constants and less sensitive to others. If a configuration assigns more digits to the sensitive constants and fewer to those less sensitive ones, we can get a smaller MSE. It is not easy to know which constants are sensitive, however, they could be searched by examining all possible configurations of a given length and compute the MSEs. Then we can find an optimized configuration with the smallest MSE. However, this exhaustive approach could be infeasible especially when there are many constants. The algorithm outlined below aims at finding an optimized configuration.

Algorithm for finding an optimized configuration with length *L* for a DCT algorithm having *n* constants

```
1. Let $P=(d_1, d_2, ...d_n)$ be the initial signed digits configuration and its
   length is $L$; $e$ = MSE of algorithm with configuration $P$; /* Number
   of additions contributed by configuration $P$ would be $L-n$. Assume
   the target complexity of the algorithm allows only $L-n$ additions
   from the signed digits configuration. */
2. for each $i \in \{1...n\}$ ; $Q_i = P$; increment $d_i$ of $Q_i$ by 1; compute MSE
   $e_{P,i}$ of algorithm with $Q_i$;   // no. of additions from $Q_i = L-n+1$.
3. find the minimum $e_{P,i}$ ; let $Q = Q_i$;
4. for each $i \in \{1...n\}$ ; let $Q_i=Q$;  decrement $d_i$ of $Q_i$ by 1; compute
   MSE $e_{Q,i}$ of algorithm with $Q_i$; // no. of additions from $Q_i$ : $L-n$
5. find the minimum $e_{Q,i}$ ; let $P = Q_i$;
6. if ($e$ not equal to $e_{Q,i}$) then {e= $e_{Q,i}$ ; goto 2;}
7. exit  // minimum MSE: $e$; optimized configuration: $P$ with length $L$
```

Fig.3 shows the MSEs of the approximated AAN's forward transform algorithm with optimized configurations of different lengths and those for configurations with equal number of digits assigned to each constant. Clearly using an optimized configuration can achieve a substantial improvement in MSE especially when the configuration length is small.

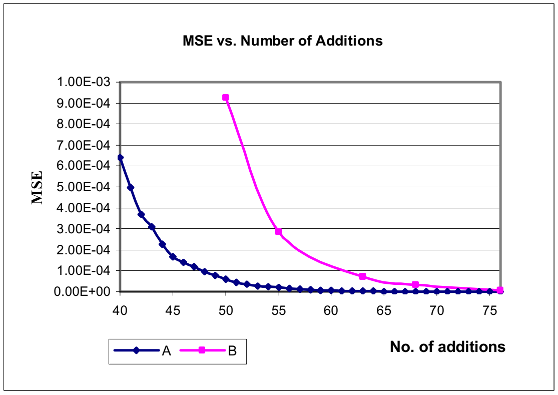

Fig. 3. MSEs of AAN's forward transform algorithm with different signed digits configurations for various target complexities (A: optimized configurations; B: configurations with equal number of digits for each constant)

5. RESULTS

The performance of different approximate multiplierless DCT algorithms can be evaluated based on three different types of results, namely, the mean square error (MSE), the algorithm complexity, and the PSNR.

5.1. Algorithm Complexity and MSE

The complexity of an approximate multiplierless DCT algorithm can be measured in terms of the number of additions and the number of shifts. Table 1 shows the complexities of the different configurations of AAN's algorithm with approximated forward transform. The complexity depends on the number of digits assigned to each constant, whether the normalization step has been converted into add-and-shift operations. If the normalization step has not been converted, there will be 8 float multiplications.

TABLE 1: THE COMPLEXITIES OF DIFFERENT CONFIGURATIONS OF THE CONVERTED AAN'S DCT ALGORITHM

Config. No.	*Description of constants*	*Norm. multi-plications*	*No. of Adds*	*No. of Shifts*
#1	*All 2 digits, float norm.*	*8*	*34*	*5*
#2	*All 4 digits, float norm.*	*8*	*44*	*15*
#3	*All 2 digits, converted norm.*	*0*	*42*	*13*
#4	*All 4 digits, converted norm.*	*0*	*68*	*39*

The MSE, defined in [3], can be considered as a kind of approximation error of a converted DCT algorithm. We note that the MSE of an approximated algorithm can be affected by its complexity measured by the number of additions. In general, a higher complexity could result in a lower MSE. Further, using more bits/digits to approximate a constant can increase the number of additions, resulting in a higher complexity of the algorithm.

5.2. Peak Signal to Noise Ratio

A common metric used to estimate the quality of a reconstructed image compared with the original image is the peak signal-to-noise ratio (PSNR), which is defined as

$$PSNR = 10\log_{10}\left(255^2 \Big/ \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}[\alpha_{(i,j)} - \beta_{(i,j)}]^2\right) \text{ in dB}$$

for any two $m \times n$ images, where $\alpha_{(i,j)}$ and $\beta_{(i,j)}$ denote respectively the original image pixel value and the reconstructed image pixel value at position (i, j).

TABLE 2: Comparison of PSNRs (dB) of approximated AAN's DCT Algorithm with different configurations

Configuration	*Non-Optimized Config.*			*Optimized config.*		
Image/No of +	*All 2*	*All 3*	*All 4*	*42**	*55***	*68****
Lena	24.36	36.73	46.25	44.81	46.06	46.11
Baboon	24.17	36.54	46.09	41.64	45.62	45.98
Peppers	24.46	36.80	46.16	44.69	46.05	46.07
Girl	38.02	48.80	47.76	47.62	46.78	46.78
Boat	23.77	36.07	45.75	43.16	45.90	46.04
Camera man	24.03	36.08	45.69	41.47	45.72	46.14
Saturn	28.00	39.55	46.86	46.07	47.13	47.19

where the constants are $\{\rho_1, \mu_1, \rho_2, C_{4a}, C_{4b}, n_0, n_1, n_2, n_3, n_4, n_5, n_6, n_7\}$

* (2,2,2,2,3,4,2,1,2,1,1,2,2), ** (3,3,3,4,4,5,3,2,3,2,2,2,3)

*** (4,3,4,5,5,5,5,4,3,4,3,3,4).

All 2, *All 3*, and *All 4* have complexities 42, 55, and 68 respectively.

Table 2 shows the PSNRs of the AAN's DCT algorithm with our proposed multiplierless forward transform and normalization based on different signed digits configurations for seven commonly used images. When the complexities are small, an optimized signed digits configuration could achieve a significant improvement in PSNR when compared to a non-optimized configuration.

6. CONCLUDING REMARKS

This paper presents an effective method to convert any float 1-D DCT transform into an approximate multiplierless version with only shift-and-add operations. Using the proposed method, we have converted AAN's fast DCT algorithms into their multiplierless versions. As AAN's fast algorithms do not have the required butterfly structures, they cannot be approximated by other published methods. The approximation error of a multiplierless algorithm can be minimized by exploiting two algorithms proposed in this paper: (1) the algorithm for converting any constant into its MSD representation (minimum number of signed digits); (2) the algorithm for finding an optimized signed digits configuration to minimize the MSE of an approximated algorithm. The proposed algorithm for finding an MSD representation of a constant is efficient and gives a signed digit string which could be shorter than a CSD representation. While an optimized configuration found by the proposed algorithm can minimize the approximation error of the multiplierless algorithm in terms of MSE, it can also improve the algorithm's performance in PSNR. Experiment results showed that the approximated AAN's DCT algorithm based on an optimized configuration could achieve a high performance in PSNR.

We will explore exploiting the proposed methodology to approximate 2-D and 3-D DCT algorithms.

7. REFERENCES

[1] K. R. Rao, and P. Yip, "Discrete Cosine Transform", *Academic Press*, pp. 82, 1990.

[2] W.K. Cham, "Development of Integer cosine transforms by the principle of dyadic symmetry," *IEE Proceedings*, Vol 135, No. 4, pp.276-282, August 1989.

[3] Trac. D. Tran, "The BinDCT: Fast Multiplierless approximation of the DCT," submitted to *IEEE Signal Processing Letters*, October 11, 1999.

[4] Jie Liang, Trac. D. Tran, "Fast Multiplierless approximations of the DCT with the lifting scheme," *IEEE Transactions on Signal Processing*, Vol 49, No. 12, pp. 3032-3044, Dec 2001.

[5] Y.J. Chen, S. Oraintara, T.D. Tran, K. Amaratunga, T.Q. Nguyen, "Multiplierless Approximation of Transforms with adder constraint," Submitted to *IEEE Signal Processing Letters*, July 2002.

[6] S.C. Chan and P.M.Yiu, "Multiplier-less Discrete Sinusoidal and Lapped Transforms using sum-of-power-of-two (SOPOT) coefficients," *Proceedings of IEEE ISCAS*, Vol 2., pp13-16, 6-9 May 2001

[7] Z. Wang, "Fast Algorithms for the Discrete W Transform and for Discrete Fourier Transform," *IEEE Trans. on ASSP*., pp. 803-816, Vol. 32 no. 4, Aug 1994.

[8] Arai Y., Agui T., Nakajima M, "A fast DCT-SQ Scheme for Images," *Trans IEICE #71*, pp.1095-1097, 1988.

[9] B.G. Lee, "A new algorithm to compute the discrete cosine transform," *IEEE Transactions on Acoustic, Speech, and Signal Processing*, vol. 32, pp. 1243-1245, Dec. 1984.

[10] G.W. Reitwiesner, "Binary arithmetics," *Advances in Computers*, 1:pp.231-308, 1960.

[11] W. Chen, C.H. Smith, and S.C. Fralick, "A fast computational algorithm for the discrete cosine transform," *IEEE Transactions on Communication*, Vol. 25, pp. 1004-1009, Sept. 1977.

[12] C. Loeffler, A. Lightenberg, and G. Moschytz, "Practical fast 1-D DCT algorithms with 11 multiplications," *Proc. IEEE ICASSP*, vol. 2, pp.988-991, Feb 1989.