China 1991 International Conference on Circuits and Systems, June 1991, Shenzhen, China

Symmetric Realization of DCT for Multi-dimensional Data Compression

Yuk-Hee Chan and Wan-Chi Siu

Department of Electronic Engineering
Hong Kong Polytechnic
Hung Hom, Kowloon, Hong Kong

ABSTRACT

The row-column approach is widely used for the realization of multi-dimensional DCT due to its simple and flexible structure. In this paper, we propose a new DCT algorithm which is most suitable for the realization of multi-dimensional DCT with the row-column decomposition technique. By using this new algorithm, the saving in terms of computational complexity is exponentially proportional to the degree of dimension compared with any other 1-D DCT algorithms. On the other hand, this new algorithm can also achieve additional saving in terms of the number of multiplications when the complete DCT realization is not necessary. These two important properties make the algorithm very suitable for the realization of multi- dimensional DCT for image compression.

Introduction

Image coding[1-3] is widely used in reducing image transmission channel bandwidth, transmission time and storage requirements to save both time and cost. Several efficient coding techniques such as predictive coding, transform coding, contour coding and hybrid coding have been developed in past years.

In transform coding, an image is transformed to a domain significantly different from the image intensity domain, and the transform coefficients are then coded. It is generally accepted that Karhunen-Loeve transform (KLT)[2] is the optimum transform. However, there is no general algorithm that enables its fast computation. Some suboptimal transforms such as the Discrete Fourier Transform (DFT), the Discrete Cosine Transform (DCT)[4] and the Hadamand transform are of some interest. Among them, the performance of the DCT is very close to that of the KLT as its basic functions are similar to that of the KLT. Besides, there are algorithms[6-11] that enable the fast computation of the DCT. These make the DCT become an important tool for transform coding in image compression. Furthermore, it is always desirable to have faster and more efficient algorithms for the computation of DCT, especially for low cost and practical applications.

A number of fast DCT algorithms have been proposed since the introduction of the DCT by Ahmed[4] et al. However, most fast DCT algorithms[6-9] are developed for the realization of one-dimensional DCT. To realize a 2-D DCT, the row-column decomposition approach is generally used. In such case, the 2-D DCT is realized by applying the 1-D DCT along each dimension. Apparently, this approach is not as efficient as some dedicated 2-D DCT algorithms[10, 11] in terms of the number of multiplications. However, it is generally used due to its simplicity in structure. Typically, row-column approach is more structural and flexible compared with those dedicated 2-D algorithms. The implementation of row-column approach is straight-forward. Its simple structure avoids complicated data management and lowers storage requirement. Besides, the row-column approach can be easily generalized and then extended to realize multi-dimensional DCT while this is hardly possible for those dedicated 2-D DCT algorithms to achieve.

On the other hand, most DCT algorithms[6-11] are proposed for the complete realization of a DCT. In other words, no additional saving in the number of mathematical operations can be achieved even through we know in advance that some DCT coefficients are of no use. However, this case is practically possible especially when the DCT is applied for image compression[3]. For example, only the coefficients within a specified region are coded in zonal image transform coding. However, by applying conventional 2-D DCT algorithms, one can hardly save much computational effort by making use of this property. Hence, it is desired that there exists an algorithm which can achieve additional saving when a complete DCT realization is not necessary.

In this paper, we propose a new DCT algorithm which is most suitable for the realisation of a multi-dimensional DCT when row-column decomposition technique is used. Compared with Lee's[8] and Hou's[9] algorithms, which are regarded as the algorithms required the minimum number of multiplications to realise a 1-D DCT, this new algorithm requires the same number of multiplications and a slightly larger number of additions. However, this new algorithm is much better than any other 1-D algorithms[6-9] in terms of the number of multiplications when they are applied to realise a multi-dimensional DCT using the row-column decomposition technique. The saving in terms of computational complexity is exponentially proportional to the degree of dimension.

Furthermore, the proposed new algorithm can achieve an additional saving in computational effort when a complete DCT realization is not necessary. In such case, the additional saving in number of multiplications is directly proportional to the number of unnecessary coefficients.

Symmetric Cosine Structure

We firstly define an N-length symmetric cosine structure, SCS[5], on a sequence $\{x(i):i=0,1..N-1\}$ as the following:

$$T(k)=\sum_{i=0}^{N-1} x(i)\cos\left(\frac{\pi ik}{N}\right) \qquad \text{for } k=0,1...N-1 \quad (1)$$

If N is even, we have

$$T(2k) = \sum_{i=0}^{\frac{N}{2}-1} g(i) \cos\left(\frac{2\pi ik}{N}\right) + (-1)^k x(\tfrac{N}{2})$$

for $k = 0,1..N/2\text{-}1$ (2)

$$\text{where} \begin{cases} g(0) = x(0) \\ g(i) = x(i) + x(N-i) \end{cases}$$

for $i = 1,2..N/2\text{-}1$ (3)

and, if we define T(-1) as T(1), we can also define the following to take care of odd terms,

$$T(2k+1) + T(2k\text{-}1) = F(k) = \sum_{i=0}^{\frac{N}{2}-1} e(i) \cos\left(\frac{2\pi ik}{N}\right)$$

for $k = 0...N/2\text{-}1$ (4)

$$\text{where} \begin{cases} e(0) = 2\ x(0) \\ e(i) = 2\ \{x(i) - x(N-i)\} \cos\left(\frac{\pi i}{N}\right) \end{cases}$$

for $i = 1,2..N/2\text{-}1$ (5)

The structure of eqn.2 will be used recursively. Hence, in order to simplify the mathematical complexity, we now give a second version of the symmetric cosine structure denoted by SCS*. Let us define an N-length SCS* on $\{x'(i):i = 0,1...N\text{-}1\}$ as follows:

$$T'(k) = \sum_{i=0}^{N-1} x'(i) \cos\left(\frac{\pi ik}{N}\right) + (-1)^k x_0$$

for $k = 0,1...N\text{-}1$ (6)

where x_0 is any real number such that eqn.6 is in the same form as eqn.2. If N is even, we have

$$T'(2k) = \sum_{i=0}^{\frac{N}{2}-1} g'(i) \cos\left(\frac{2\pi ik}{N}\right) + (-1)^k x'(\tfrac{N}{2})$$

for $k = 0,1...N/2\text{-}1$ (7)

$$\text{where} \begin{cases} g'(0) = x'(0) + x_0 \\ g'(i) = x'(i) + x'(N-i) \end{cases}$$

for $i = 1,2..N/2\text{-}1$ (8)

and, similarly, if we define T'(-1) as T'(1), we have

$$T'(2k+1) + T'(2k\text{-}1) = F'(k) = \sum_{i=0}^{\frac{N}{2}-1} e'(i) \cos\left(\frac{2\pi ik}{N}\right)$$

for $k = 0,1..N/2\text{-}1$ (9)

$$\text{where} \begin{cases} e'(0) = 2\ \{x'(0) - x_0\} \\ e'(i) = 2\ \{x'(i) - x'(N-i)\} \cos\left(\frac{i\pi}{N}\right) \end{cases}$$

for $i = 1,2..N/2\text{-}1$ (10)

Hence, both an N-length SCS and an N-length SCS* can be decomposed into an N/2 length SCS and an N/2-length SCS* with a cost of N/2-1 multiplications. The mathematical complexity of a 2^m-length SCS and 2^m- length SCS* is given by the following sets of recursive equations respectively:

$$M(N-SCS) = M(\tfrac{N}{2}-SCS) + M(\tfrac{N}{2}-SCS^*) + \tfrac{N}{2} - 1$$
$$A(N-SCS) = A(\tfrac{N}{2}-SCS) + A(\tfrac{N}{2}-SCS^*) + \tfrac{3N}{2} - 3$$
$$M(N-SCS^*) = M(\tfrac{N}{2}-SCS) + M(\tfrac{N}{2}-SCS^*) + \tfrac{N}{2} - 1$$
$$A(N-SCS^*) = A(\tfrac{N}{2}-SCS) + A(\tfrac{N}{2}-SCS^*) + \tfrac{3N}{2} - 1$$

where $N = 2^m$ and $m > 2$

and

$$M(4-SCS) = 1$$
$$A(4-SCS) = 7$$
$$M(4-SCS^*) = 1$$
$$A(4-SCS^*) = 9 \qquad (11)$$

where M(n-Y) and A(n-Y) are the numbers of multiplications and additions respectively for a length-n structure denoted by Y.

Realization of 1-D DCT

Recall that an N-length DCT[4] on sequence $\{y(i):i = 0,1..N\text{-}1\}$ is given by

$$Y(k) = \sum_{i=0}^{N-1} y(i) \cos\left(\frac{\pi(2i+1)k}{2N}\right)$$

for $k = 0,1...N\text{-}1$ (12)

By some simple additions, we have another sequence $\{x(i):i = 0,1..N\text{-}1\}$ such that

$$\begin{cases} x(0) = x'(0) \\ x(i) = 2x'(i) \end{cases} \quad \text{for } i = 1,2..N\text{-}1 \quad (13)$$

$$\text{where} \begin{cases} x'(N-1) = y(N-1) \\ x'(i) + x'(i+1) = y(i) \end{cases}$$

for $i = 0,1..N\text{-}2$ (14)

Then it can be shown that Y(k) can be obtained by the realization of the N-length SCS on input sequence x(i) with an additional cost of N-1 multiplications and N-1 additions:

$$Y(k) = \left\{ \sum_{i=0}^{N-1} x(i) \cos\left(\frac{\pi ik}{N}\right) \right\} \cos\left(\frac{\pi k}{2N}\right)$$

for $k = 0,1..N\text{-}1$ (15)

The mathematical complexity of this new algorithm to realize the DCT is given by the following recursive equations:

$$M(N\text{-}DCT) = M(N\text{-}SCS) + N - 1$$
$$A(N\text{-}DCT) = A(N\text{-}SCS) + N - 1$$

where $N = 2^m$ (16)

In an non-recursive form, we have

$$M(N-DCT) = \tfrac{N}{2} \log_2 N \qquad (17)$$

which gives the same minimum number of multiplications as that required by Lee's algorithm[8]. Table 1 shows a comparison between this new algorithm and some selected DCT algorithms[6,8,9].

N	New Algorithm		Lee Algorithm		Hou Algorithm		Chen Algorithm	
	M	A	M	A	M	A	M	A
4	4	10	4	9	4	9	6	8
8	12	32	12	29	12	29	16	26
16	32	88	32	81	32	81	44	74
32	80	224	80	209	80	209	116	194
64	192	544	192	513	192	513	292	482
128	448	1280	448	1217	448	1217	708	1154

Table 1. Comparison of mathematical complexity between this new algorithm and selected DCT algorithms[6,8,9].

Realization of 2-D DCT

Recall that a 2-dimensional N_1*N_2 DCT on $\{y(i,j): i = 0,1..N_1\text{-}1; j = 0,1..N_2\text{-}1\}$ is defined as

$$Y(k,q) = \sum_{j=0}^{N_2-1} \sum_{i=0}^{N_1-1} y(i,j) \cos\left(\frac{(2i+1)k\pi}{2N_1}\right) \cos\left(\frac{(2j+1)q\pi}{2N_2}\right)$$

for $k = 0,1..N_1\text{-}1$; $q = 0,1..N_2\text{-}1$ (18)

Typically, we can realize this 2-D DCT by a row-column decomposition. This approach is assumed to be the most standard and simplest one and it can be realized easily due to its regular structure. We find that the new algorithm shows its strength on realizing 2-D DCT when row-column decomposition approach is used.

Let us define another sequence $\{x(i,j):i=0,1..N_1-1;\ j=0,1..N_2-1\}$

$$\begin{cases} x(0,j) = x'(0,j) & \\ x(i,j) = 2\,x'(i,j) & \text{for } i=1,2...N_1-1 \end{cases}$$

$$\begin{cases} x'(N_1-1,j) = y(N_1-1,j) & \\ x'(i,j) = y(i,j) - x'(i+1,j) & \text{for } i=0,1...N_1-2 \end{cases}$$

$$\text{both for } j=0,1...N_2-1 \quad (19)$$

Then we have

$$\sum_{i=0}^{N_1-1} y(i,j)\cos\left(\frac{(2i+1)k\pi}{2N_1}\right) = \sum_{i=0}^{N_1-1}\left\{x(i,j)\cos\left(\frac{ik\pi}{N_1}\right)\right\}\cos\left(\frac{k\pi}{2N_1}\right)$$

$$\text{for } k=0,1..N_1-1;\quad j=0,1..N_2-1 \quad (20)$$

such that

$$Y(k,q) = \sum_{j=0}^{N_2-1} f(j,k)\cos\left(\frac{(2j+1)q\pi}{2N_2}\right)\cos\left(\frac{k\pi}{2N_1}\right)$$

$$\text{for } k=0,1..N_1-1;\quad q=0,1..N_2-1 \quad (21)$$

$$\text{where } f(j,k) = \sum_{i=0}^{N_1-1} x(i,j)\cos\left(\frac{ik\pi}{N_1}\right)$$

$$\text{for } k=0,1..N_1-1;\quad j=0,1..N_2-1 \quad (22)$$

By using a similar technique as above, we have

$$Y(k,q) = \left\{\sum_{j=0}^{N_2-1} g(j,k)\cos\left(\frac{jq\pi}{N_2}\right)\right\}\cos\left(\frac{q\pi}{2N_2}\right)\cos\left(\frac{k\pi}{2N_1}\right)$$

$$\text{for } k=0,1..N_1-1;\quad q=0,1..N_2-1 \quad (23)$$

$$\text{where } \begin{cases} g(0,k) = f'(0,k) & \\ g(j,k) = 2\,f'(j,k) & \text{for } j=1,2...N_2-1 \end{cases}$$

$$\begin{cases} f'(N_2-1,k) = f(N_2-1,k) & \\ f'(j,k) + f'(j+1,k) = f(j,k) & \text{for } j=0,1...N_2-2 \end{cases}$$

$$\text{both for } k=0,1...N_1-1 \quad (24)$$

In short, we can realize this 2-D DCT of eqn.(18) by the following steps:

I. compute $\{x(i,j)\}$ from $\{y(i,j)\}$ for $i=0,1..N_1-1$ and $j=0,1..N_2-1$;

II. perform 1-D SCSs on $\{x(i,j)\}$ for each j, where $j=0,1...N_2-1$;

III. compute $\{g(j,k)\}$ from $\{f(j,k)\}$ for $k=0,1..N_1-1$ and $j=0,1..N_2-1$;

IV. perform 1-D SCSs on $\{g(j,k)\}$ for each k, where $k=0,1...N_1-1$;

V. multiply $\cos(\pi q/2N_2)\cos(\pi k/2N_1)$ accordingly.

Note that $\cos(\pi q/2N_2)\cos(\pi k/2N_1)$ terms can be precomputed and stored in memory such that we can retrieve them by the table-lookup technique. If $N_1=N_2=N=2^m$, then as $\cos(\pi q/2N)\cos(\pi k/2N)$ is symmetric on k and q, only a memory size of $(N+1)N/2-1$ is required for the table.

As the multiplication of $\cos^2(\pi/4)$ and $\cos^2(0)$ are trivial, the multiplicative complexity of an N*N 2-D DCT can then be given by

$$M(N{*}N\text{-DCT}) = 2N{*}M(N\text{-SCS}) + N^2 - 2$$
$$A(N{*}N\text{-DCT}) = 2N{*}\{A(N\text{-SCS}) + N - 1\} \quad (25)$$

Table 2 shows a comparison of the multiplicative complexity of an N*N DCT between the new algorithm and some selected algorithms[8,9]. As the row-column decomposition technique is assumed to be used, M(N*N-DCT) = 2N*M(N-DCT) and A(N*N-DCT) = 2N*A(N-DCT) when selected algorithms[8,9] are used to realize a 2-D DCT. This table shows that the present approach requires significantly less multiplications compared with all other techniques whereas the number of additions of the present approach is slightly larger than other techniques. For an example, for the realization of a 16*16 DCT, the present approach requires 22% less multiplications, and only 8% more additions compared with the corresponding operations required by Lee's or Hou's approach. We may appreciate these figures better if we recall the fact that a multiplication is usually more expensive than an addition. Furthermore, as shown in the table, the present approach requires the least number of total operations compared with all other techniques. Again we have to emphasize that, due to their simplicity in structure, approaches[6-9] suitable for row-column decomposition technique are used mainly for our comparison. Other dedicated 2-D approaches[10,11] which might appear to require less numbers of operations for their realization are not considered here since their structures are usually irregular, relatively complicated and difficult to be realised.

In a practical application on image data compression, the DCT acts as a great tool in transform image coding. As transform image coding exploits the energy compaction property, only a small fraction of the transform coefficients is required to be coded. There are two approaches commonly used to determine whether a transform coefficient should be coded: zonal coding and threshold coding. In zonal coding approach, only the coefficients within a specified region are coded. This region can be predefined. However, by making use of conventional DCT algorithm, we cannot save much mathematical effort by making use of this property. In contrast to this, the proposed new algorithm can make use of this property to reduce a number of multiplications as it is not necessary to multiply $\cos(\pi q/2N)\cos(\pi k/2N)$ to the corresponding discarded transform coefficients. Consider a typical case that half of the transform coefficients are discarded during coding, then we can save $N^2/2$ multiplications. Column 2 of table 2 shows the case with additional savings on a 2-D N*N DCT realization based on this consideration.

$N_1{*}N_2$	New Algorithm		New Algorithm #		Lee Algorithm		Hou Algorithm	
	M	A	M	A	M	A	M	A
4*4	22	80	14	80	32	72	32	72
8*8	142	512	110	512	192	464	192	464
16*16	798	2816	670	2816	1024	2592	1024	2592
32*32	4158	14336	3646	14336	5120	13376	5120	13376

Table 2. Comparison of mathematical complexity between this new algorithm and selected DCT algorithms[8,9] using the row-column decomposition technique. (In new algorithm#, half of transform coefficients are discarded.)

This new algorithm has been realized using a 286 personal computer for the sake of comparison with selected algorithms[8,9]. Instead of realizing equation (18), we realize

$$Y(k,q) = m(k,q)\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} y(i,j)\cos\left(\frac{(2i+1)k\pi}{2N}\right)\cos\left(\frac{(2j+1)q\pi}{2N}\right)$$

$$\text{for } k,q=0,1..N-1 \quad (26)$$

$$\text{where } m(k,q) = \begin{cases} (1/N)^2 & k=q=0 \\ 2\,(1/N)^2 \quad \text{for} & k=0 \text{ or } q=0,\ k\neq q \\ 4\,(1/N)^2 & k\neq 0,\ q\neq 0 \end{cases}$$

as it is more commonly used in image transform coding. Table 3 shows a comparison of its computation speeds to that of row-column approach with Lee's algorithm[8] or Hou's algorithm[9]. All programmes were written in Intel 80286 assembly language and have been executed on an IBM PC/AT compatible machine without 80287 Math coprocessor. All algorithms were realized based on integer multiplications and additions only. Original floating point data were scaled in order to improve the accuracy and to be able to use integer arithmetic.

N*N	New Algorithm*	New Algorithm	Lee Algorithm	Hou Algorithm
4*4	0.577ms	0.620ms	0.889ms	1.269ms
8*8	3.265ms	3.457ms	4.903ms	6.757ms
16*16	17.180ms	18.020ms	24.880ms	34.220ms

Table 3. Results of a comparison between the new algorithm and selected algorithms[8,9].
(In new algorithm*, transform coefficients $\{T_c(i,j) : i+j > N\}$ are discarded.)

As Lee's algorithm[8] requires the division of the cosine coefficients, it causes numerical instabilities because of roundoff errors in finite length registers. To avoid the occurrence of overflow, the scale factor used in the realization of Lee's algorithm is a little smaller than that used in the new algorithm, which lowers the accuracy of the output of the Lee's algorithm[8]. On the other hand, Hou's algorithm[9] involves a number of bit-reversal operations during the realization of DCT, which are very time-consuming and affect its performance significantly. Table 3 shows the results of our actual realization and it is clear from the figure that our approach always gives the best timing.

N-Dimensional DCT

By using a similar technique as the row-column decomposition approach, we can extend the new algorithm to a multi-dimensional case. The mathematical complexity of a k-dimensional, N*N..*N-DCT is given by:

$$\begin{aligned} M(N^k\text{-DCT}) &= kN^{k-1} M(N\text{-SCS}) + N^k - 1 - \{1+(-1)^k\}/2 \\ A(N^k\text{-DCT}) &= kN^{k-1} \{ A(N\text{-SCS}) + N - 1 \} \end{aligned} \qquad (27)$$

By making a comparison with the row-column-decomposition approach for realizing Lee's and Hou's algorithm[8,9], the number of multiplications saved is $(k-1)N^k-kN^{k-1}+1+\{1+(-1)^k\}/2$ which increases amazingly as k increases.

This theoretical result can be applied to transform image coding too. On the assumption that the intensity of an image pixel along the time axis is correlated to subsequent video pictures, we can define a bijective mapping such that we can project a 3-D data p(x,y,t) to a k-D data $p(x,y,t_1,t_2,..t_{k-2})$. Then we can perform a k-D DCT on $p(x,y,t_1,..t_{k-2})$ instead of a 2-D DCT on p(x,y). This will save a lot of multiplications and will speed up the transform process. Unfortunately, this approach requires a large memory size to store up a number of frames during inverse transform. However, it may not be impractical if k is small while the cost of memory is falling.

Conclusions

In this paper, the symmetric cosine structure is introduced. Based on the fast algorithm proposed to realize a 2^m-length SCS, we propose a new algorithm to realize a 2^m-length discrete cosine transform. This new algorithm has a number of advantages compared with other existing algorithms. For the one-dimensional DCT case, it requires the same minimum number of multiplications reported in the literature[7,8,9], whereas for the realization of two-dimensional DCT, it requires the least number of operations compared with other fast algorithms[6-9] that are suitable for row-column decomposition. It is significant to point out that this saving is due mainly to a reduction in the number of multiplications which are usually more expensive compared with the cost of additions. The number of multiplications can be further reduced by the fact that a number of transform coefficients can be discarded during the coding in the Zonal coding method for image compression. This enables us to give the best timing performance for image compression using the present approach. This new approach can be extended to a multi-dimensional case, in which the saving in terms of computational complexity is exponentially proportional to the degree of dimension compared with row-column approach using any other 1-D DCT algorithms[6-9]. Lastly, it is worth to mention that this new approach is numerically stable, as it involves the multiplication of cosine terms only and the structure is flexible and regular. This last point suggests that the new approach should also be suitable for VLSI or gate array realization.

References

[1] A.N.Netravali and J.O.Limb, "Picture Coding:A Review," Proceedings of the IEEE, Vol.68, No.3, pp.366-406, Mar. 1980.

[2] A.K.Jain, "Image data compression: A review," Proceedings of the IEEE, Vol.69, No.3, pp.349-389, Mar. 1981.

[3] J.S.Lim, *Two-dimensional Signal and Image Processing*, Prentice-Hall International Editions, 1990, pp.524-575.

[4] N.Ahmed, T. Natarajan and K.R.Rao, "Discrete cosine transform," IEEE trans., Vol.C-23, pp.90-94, Jan. 1974.

[5] Y.H.Chan and W.C.Siu, "Modular Structures for Fast Discrete Hartley Transform and Real-valued Discrete Fourier Transform," Proceedings, ISSPA'90, 27-31 August, 1990, Gold Coast, Australia, pp.221-224.

[6] W.H.Chen, C.H.Smith and S.C.Fralick, "A fast computational algorithm for the discrete cosine transform," IEEE Trans., Vol.COM-25, pp.1004- 1009, Sep. 1977.

[7] M.Vetterli and H.Nussbaumer, "A simple FFT and DCT Algorithms with Reduced Number of Operations," Signal Processing, Vol.6, pp.267-278, 1984.

[8] B.G.Lee, "A new algorithm to compute the discrete cosine transform," IEEE Trans., Vol.ASSP-32, pp.1243-1245, Dec. 1984.

[9] H.S.Hou, "A fast recursive algorithm for computing the discrete cosine transform," IEEE Trans., Vol.ASSP-35, pp.1455-1461, Oct. 1987.

[10] M.A.Haque, "A two dimensional fast cosine transform," IEEE Trans., Vol.ASSP-33, pp.1532-1539, Dec. 1985.

[11] F.A.Kamangar and R.Rao, "Fast Algorithms for the 2-D Discrete Cosine Transform," IEEE Trans on Computers, Vol.C-31, No.9, Sep. 1982.