Hindawi
Mathematical Problems in Engineering
Volume 2017, Article ID 7401845, 7 pages
https://doi.org/10.1155/2017/7401845

Research Article

A Fast DCT Algorithm for Watermarking in Digital Signal Processor

S. E. Tsai[1] and S. M. Yang[2]

[1]*Department of Computer Science and Information Engineering, Chang Jung Christian University, Tainan City 701, Taiwan*
[2]*Department of Aeronautics and Astronautics, National Cheng Kung University, Tainan City 701, Taiwan*

Correspondence should be addressed to S. M. Yang; smyang@mail.ncku.edu.tw

Received 3 November 2016; Accepted 23 January 2017; Published 9 February 2017

Academic Editor: Xinkai Chen

Discrete cosine transform (DCT) has been an international standard in Joint Photographic Experts Group (JPEG) format to reduce the blocking effect in digital image compression. This paper proposes a fast discrete cosine transform (FDCT) algorithm that utilizes the energy compactness and matrix sparseness properties in frequency domain to achieve higher computation performance. For a JPEG image of 8×8 block size in spatial domain, the algorithm decomposes the two-dimensional (2D) DCT into one pair of one-dimensional (1D) DCTs with transform computation in only 24 multiplications. The 2D spatial data is a linear combination of the base image obtained by the outer product of the column and row vectors of cosine functions so that inverse DCT is as efficient. Implementation of the FDCT algorithm shows that embedding a watermark image of 32×32 block pixel size in a 256×256 digital image can be completed in only 0.24 seconds and the extraction of watermark by inverse transform is within 0.21 seconds. The proposed FDCT algorithm is shown more efficient than many previous works in computation.

1. Introduction

Discrete cosine transform (DCT) has been widely used to convert a dynamic signal into frequency components so as to reduce digital image storage size, expedite data transmission, and remove redundant information. DCT is closely related to discrete Fourier transform with the advantage of concentrating the energy of transformed signal in low frequency range where human eyes are less sensitive in image processing [1]. The joint ISO committee therefore adopts DCT to Joint Photographic Experts Group (JPEG) international standard of 8×8 block size to reduce the blocking effect in image compression. A basic JPEG image encoding is composed of three procedures: image transform, quantization, and encoding.

DCT can map an original data into frequency domain by cosine waveform, and conversely inverse discrete cosine transform (IDCT) transfers frequency domain data into spatial domain. Numerous coding methods based on DCT have been presented for digital image processing; however, the associated memory size, bandwidth, and safety issues are of significant concern to real-time applications. Sun and Yang [2] proposed an image compression method based on a Laplace transparent composite model to achieve high coding efficiency. Jridi et al. [3] presented image compression hardware to reduce computational complexity. Others have proposed to optimize image computation by digital signal processor (DSP). Kumbhare and Gokhale [4] developed a low complexity architecture for computing an algebraic integer based 8-point DCT in digital image processing. Jridi et al. [5] designed a low complexity DCT engine in digital video and image processing. Subband decomposition algorithms based on DCT have also been used in transmitting image data of low resolution to rebuilt image of better quality [6–8], but they required high complexity and thus time-consuming computation. Stassen's matrix multiplication algorithm was proposed to reduce complex matrix multiplication in DCT [9]. Khan et al. [10] increased the coordination between the pixel size and subword size to maximize resource utilization for multimedia application, but

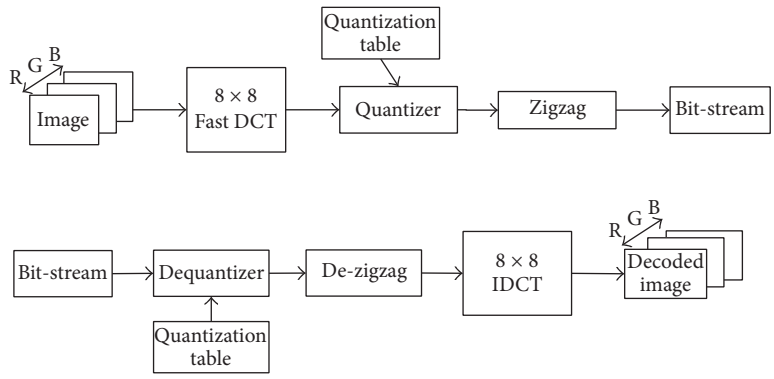

Figure 1: The encoder and decoder model of JPEG compression standard by using discrete cosine transform (DCT) and inverse discrete cosine transform (IDCT).

the work required heavy computation. This paper proposes a fast DCT (FDCT) algorithm with significantly reduced number of multiplications to achieve higher computation efficiency in digital image processing. It is also shown suitable for hardware implementation in DSP on digital watermarking applications.

2. DCT in JPEG

The basic JPEG image encoding method is composed of three procedures: image transform, quantization, and encoding. Figure 1 shows the encoder and decoder model, where an original image is first divided into block pixel size 8 × 8 in RGB model with each block in 0 to 63 frequency coefficients as shown in Figure 2. The low frequency coefficients are in the light color region. During image processing, DCT maps the spatial domain data into frequency domain by cosine waveform and conversely in inverse discrete cosine transform [11]. The spatial domain indicates the "magnitude" of a color image while the frequency domain shows the magnitude change from one pixel to the next. In DCT, the original host signal is first divided into nonoverlapping 2D blocks of size 8 × 8. Each block is then processed independently and transformed into AC and DC coefficients in frequency domain, representing the average color of the block and the color change across the block, respectively.

After DCT, a quantizer with quantization table is used to provide higher compassion ratio in transmission by approximating a continuous set of values in image data to a finite (preferably small) set of values. It is done by dividing each component in frequency domain by a constant and then rounding to the nearest integer. The input to a quantizer is the original data and the output is by a function of a set of discrete, finite output values. A good quantizer is to represent the original signal with minimum loss or distortion. A highly useful feature of JPEG process is that varying levels of image compression and quality are obtained by the selection of specific quantization matrix, similar to weighting function to mean psychological visual capability. Quantization involves dividing each coefficient by an integer value between 1 and 255.

After quantization, the DC coefficient, which contains a significant fraction of the total image energy, becomes a measure of the average value of the original 64 pixels, and the 63 AC components are treated in an entropy coding process in the order of increasing frequency. Because the 8 × 8 blocks are usually with strong correlation, the quantized DC coefficient is encoded as the difference from the DC term of the previous block. The higher frequency coefficients are more likely to be 0 or negligible after quantization, thereby improving the compression of run-length encoding.

3. Fast Discrete Cosine Transform (FDCT)

3.1. 1D FDCT. In proposed fast discrete cosine transform (FDCT) algorithm, an original signal is first divided into nonoverlapping 2D blocks of size 8 × 8 in three color components as shown in Figure 1 in JPEG format. Each block is then processed independently by the transform. Consider a spatial domain image data of 8-point, $\mathbf{s} = [s(0) \quad s(1) \quad \cdots \quad s(7)]^T$ being transformed into frequency domain $\mathbf{f}$, $\mathbf{f} = [f(0) \quad f(1) \quad \cdots \quad f(7)]^T$, where

$$f(k) = \frac{1}{2} C(k) \sum_{i=0}^{7} s(i) \cos\left(\frac{(2i+1)k\pi}{16}\right) \tag{1}$$

with $C(k) = 1/\sqrt{2}$, if $k = 0$, and $C(k) = 1$ for others. $f(0)$ and $f(4)$ can be evaluated without multiplication, only by addition and subtraction. There are only six elements left in $\mathbf{f}$ to be evaluated. Further minimization of the number of

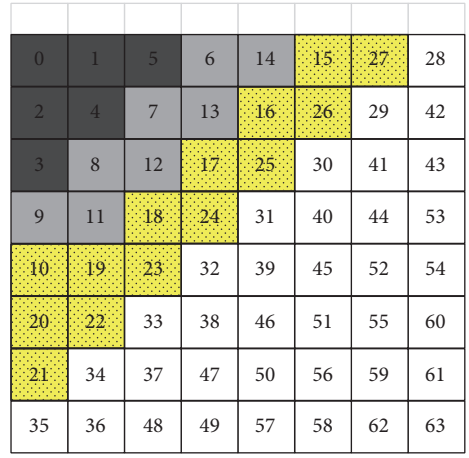

Figure 2: The distribution of coefficients in DCT, (a) low frequency position (heavy color), (b) low-middle-frequency position (light color), (c) high-middle frequency position (light color with dots), and (d) high frequency position (white).

multiplications can be achieved by regrouping the coefficients by using the symmetric property to yield

$$\begin{bmatrix} f(2) \\ f(6) \\ f(7) \\ f(5) \\ f(1) \\ f(3) \end{bmatrix} = \begin{bmatrix} b & d & 0 & 0 & 0 & 0 \\ d & -b & 0 & 0 & 0 & 0 \\ 0 & 0 & e & f & -a & c \\ 0 & 0 & f & a & c & e \\ 0 & 0 & -a & c & -e & -f \\ 0 & 0 & c & e & -f & -a \end{bmatrix} \begin{bmatrix} s(0) - s(3) - s(4) + s(6) \\ s(1) - s(2) - s(5) + s(6) \\ s(0) - s(7) \\ s(6) - s(1) \\ s(3) - s(4) \\ s(2) - s(5) \end{bmatrix}, \tag{2}$$

where $a = 0.488$, $b = 0.463$, $c = 0.416$, $d = 0.192$, $e = 0.098$, and $f = 0278$. The number of multiplications is now reduced to 20. To eliminate one more multiplication, $f(2)$ and $f(6)$ can be evaluated by

$$\begin{bmatrix} f(2) \\ f(6) \end{bmatrix} = \mathbf{A} \begin{bmatrix} (b-d)(s(0) - s(3) - s(4) + s(7)) \\ d(s(0) + s(1) - s(2) - s(3) - s(4) - s(5) + s(6) + s(7)) \\ -(b-d)(s(1) - (2) - s(5) + s(6)) \end{bmatrix}, \tag{3}$$

where $\mathbf{A} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$, so that $f(1)$, $f(3)$, $f(5)$, and $f(7)$ become

$$\begin{bmatrix} f(7) \\ f(5) \\ -f(1) \\ f(3) \end{bmatrix} = \mathbf{A}^* \begin{bmatrix} \mathbf{A} \begin{bmatrix} (e+a-f+c)(s(0) - s(7)) \\ (f-c)(s(0) - s(7) + s(6) - s(1)) \\ (a-e-f+c)(s(6) - s(1)) \end{bmatrix} \\ \mathbf{A} \begin{bmatrix} (-a-c)(s(0) - s(7) + s(3) - s(4)) \\ c(s(0) - s(7) + s(3) - s(4) + s(6) - s(1) + s(2) - s(5)) \\ (e-c)(s(6) - s(1) + s(2) - s(5)) \end{bmatrix} \\ \mathbf{A} \begin{bmatrix} (e-a-f-c)(s(3) - s(4)) \\ (f+c)(s(3) - s(4) + s(2) - s(5)) \\ (a+e-f-c)(s(2) - s(5)) \end{bmatrix} \end{bmatrix}, \tag{4}$$

where $\mathbf{A}^* = \begin{bmatrix} \mathbf{I} & \mathbf{I} \\ \mathbf{I} & -\mathbf{I} \end{bmatrix}^{0}$ with $\mathbf{I}$ being the 2×2 identity matrix. By now, DCT only needs 12 multiplications. Similarly, the inverse transform is

$$s'(i) = \frac{1}{2}\sum_{k=0}^{7} C(k)\, f(k) \cos\left(\frac{(2i+1)\,k\pi}{16}\right), \tag{5}$$

where $C(k) = 1/\sqrt{2}$, if $k = 0$, and $C(k) = 1$ for others. Careful observation reveals that it is straightforward to derive inverse transform IDCT from DCT by

$$\begin{bmatrix} s'(0) \\ s'(1) \\ s'(2) \\ s'(3) \\ s'(4) \\ s'(5) \\ s'(6) \\ s'(7) \end{bmatrix} = \begin{bmatrix} 1 & 1 & b & d & e & f & -a & c \\ 1 & -1 & d & -b & -f & -a & -c & -e \\ 1 & -1 & -d & b & c & e & -f & -a \\ 1 & 1 & -b & -d & -a & c & -e & -f \\ 1 & 1 & -b & -d & a & -c & e & f \\ 1 & -1 & -d & b & -c & -e & f & a \\ 1 & -1 & d & -b & f & a & c & e \\ 1 & 1 & b & d & -e & -f & a & -c \end{bmatrix} \begin{bmatrix} f(0) \\ f(4) \\ f(2) \\ f(6) \\ f(7) \\ f(5) \\ -f(1) \\ f(3) \end{bmatrix}. \tag{6}$$

By shifting and swapping the corresponding rows, (6) can be decomposed as

$$\begin{bmatrix} s'(0) \\ s'(6) \\ s'(3) \\ s'(2) \\ s'(7) \\ s'(1) \\ s'(4) \\ s'(5) \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} \mathbf{B}+\mathbf{C} \\ \mathbf{B}-\mathbf{C} \end{bmatrix} + \mathbf{D} \\ \begin{bmatrix} \mathbf{B}+\mathbf{C} \\ \mathbf{B}-\mathbf{C} \end{bmatrix} - \mathbf{D} \end{bmatrix}. \tag{7}$$

The matrix is made up in three building blocks with symmetric coefficient matrices,

$$\mathbf{B} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} f(0) \\ f(4) \end{bmatrix},$$

$$\mathbf{C} = \begin{bmatrix} b & d \\ d & -b \end{bmatrix} \begin{bmatrix} f(2) \\ f(6) \end{bmatrix},$$

$$\mathbf{D} = \begin{bmatrix} e & f & -a & c \\ & a & c & e \\ & & -e & -f \\ \text{symm.} & & & -a \end{bmatrix} \begin{bmatrix} f(7) \\ f(5) \\ -f(1) \\ f(3) \end{bmatrix}. \tag{8}$$

Both FDCT and fast IDCT have the same coefficients and matrix blocks. It is thus efficient in hardware implementation.

3.2. 2D FDCT. Implementation of a 2D DCT is by separating into a pair 1D DCT as illustrated in Figure 3. Consider a 2D spatial data sequence $s(i, j)$, $0 \le i, j \le 7$, in matrix $\mathbf{S}$ of 8×8, and the corresponding 2D DCT sequence $f(u, v)$, $0 \le u, v \le 7$, in frequency domain of matrix $\mathbf{F}$ is defined as

$$f(u,v) = \frac{1}{4} C(u)\, C(v) \sum_{i=0}^{7}\sum_{j=0}^{7} s(i,j) \cos\left(\frac{(2i+1)\,u\pi}{16}\right) \cdot \cos\left(\frac{(2j+1)\,v\pi}{16}\right). \tag{9}$$

The inverse transformation represented by $\mathbf{S}'$, $\mathbf{S}' = \{s'(i, j)\}$,

$$s'(i,j) = \frac{1}{4}\sum_{u=0}^{7}\sum_{v=0}^{7} C(u)\, C(v)\, f(u,v) \cdot \cos\left(\frac{(2i+1)\,u\pi}{16}\right) \cos\left(\frac{(2j+1)\,v\pi}{16}\right), \tag{10}$$

where $C(u), C(v) = 1/\sqrt{2}$ if $u, v = 0$ and $C(u)$, $C(v) = 1$ for others. By defining a matrix $\mathbf{M} = \{m(u, v)\}$, where $m(u, v)$ represents the matrix element in the uth row and vth column,

$$m(u,v) = \begin{cases} \dfrac{1}{2\sqrt{2}}, & u = 0 \cup 0 \le v \le 7 \\ \dfrac{1}{2}\cos\dfrac{(2v+1)\,u\pi}{16}, & 1 \le u \le 7 \cup 0 \le v \le 7. \end{cases} \tag{11}$$

A 2D DCT data matrix $\mathbf{F}$ and its inverse matrix $\mathbf{S}'$ can be written as

$$\mathbf{F} = \mathbf{M}\mathbf{S}\mathbf{M}^T, \tag{12a}$$

$$\mathbf{S}' = \mathbf{M}^T\mathbf{F}\mathbf{M}. \tag{12b}$$

Because the base vectors of DCT are orthogonal, the inverse transform IDCT can therefore be easily obtained as shown in

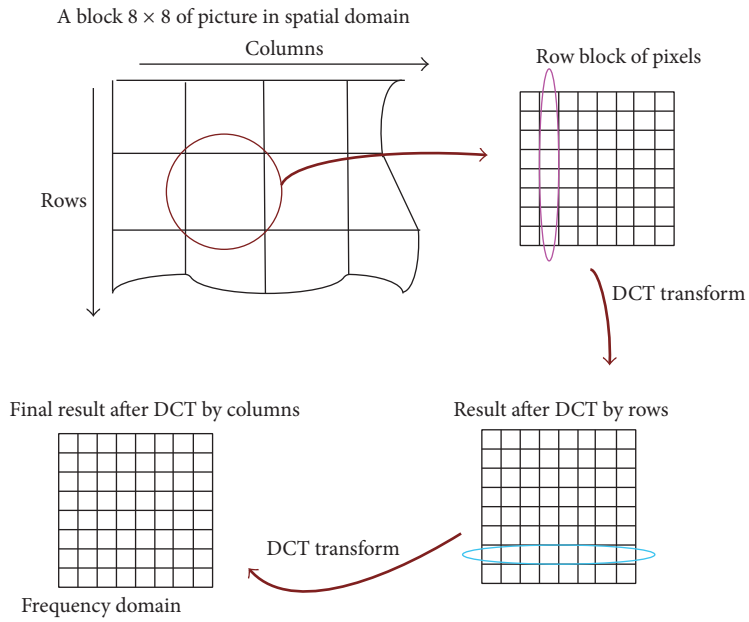

Figure 3: The FDCT algorithm of 2D DCT calculated by one pair of 1D DCTs.

(12b). DCT transforms high correlated image into a few transform coefficients. Conventional image coding techniques use the quantization process to achieve higher compression ratio. Therefore, **F** includes only a few nonzero elements in the low frequency range, which makes it possible to design efficient IDCT algorithm by fully utilizing the computation efficiency in (3) and (4) and the energy compactness property of **F**.

The 2D spatial data matrix $\mathbf{S}'$ in (12b) can be considered as linear combination of the base images or the outer product of the column and row vectors in **M**. This interpretation makes it easier for (12a) and (12b) to manipulate the sparseness of the 2D DCT matrix **F** and to calculate the spatial data matrix $\mathbf{S}'$. The proposed FDCT algorithm achieves high computation performance over many other previous algorithms. The row (or column) data can be processed by using 1D DCT (or IDCT) first with the results stored in transposition memory. By exploiting the redundancy in the coefficients of DCT, the algorithm reduces the complexity of 2D DCT of an 8×8 block to only 24 multiplications.

4. Performance Evaluation in Digital Image Watermarking

Discrete cosine transform (DCT) can map an original digital data into frequency domain by cosine waveform, and, conversely, inverse discrete cosine transform (IDCT) transfers the frequency domain data into spatial domain. The associated memory size, bandwidth, and safety issues in the transformation algorithms are of significant concern. Direct computation of 2D DCT ($N \times N$ pixel size of a block) requires N^4 multiplications, while direct realization (with the row-column separation) of 8×8 DCT $2 \times 8^3 = 1024$ multiplications. Although many algorithms have been proposed to optimize multiplications, they either required huge amount of computation [6–8, 10] or suffered from low efficiency [4, 5, 9]. Table 1 lists the number of multiplications required in preview works compared with the proposed FDCT algorithm. The latter is much more efficient in computation as described in (2) to (12a) and (12b) where only $12 \times 2 = 24$ multiplications are needed. By comparison, Jridi et al. [5] needed 256 multiplications as they simply calculated 8×8 standard DCT. Similarly, Ko et al. [7] required 31 multiplications, while Sung et al. [8] based on subband decomposition needed 25 multiplications. Manoria and Dixit [9] used Stassen's matrix multiplication but their 2D DCT (8×8) needed 28 multiplications. Khan et al. [10] took 40 multiplications for 2D DCT operation on an image of 8×8 block. In summary, both the FDCT algorithm and its inverse transform are more efficient than many

Table 1: The number of multiplications for 2D DCT by the algorithms of previous works and by the proposed FDCT algorithm.

Algorithm	Number of multiplications for 2D DCT (8×8 block)
Direct computation	$8^4 = 4{,}096$
Direct computation (with line-column separation)	$2 \times 8^3 = 1024$
Jridi et al. [5]	256
Ko et al. [7]	31
Sung et al. [8]	25
Manoria and Dixit [9]	28
Khan et al. [10]	40
FDCT algorithm (this work)	24

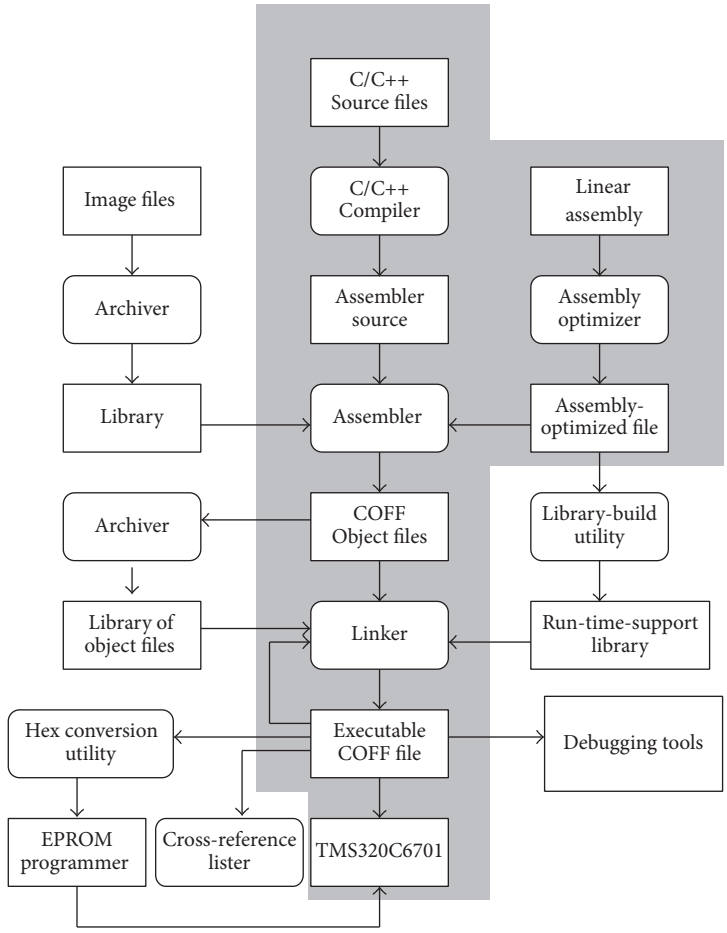

Figure 4: Software environment in implementing the FDCT algorithm on DSP (TMS320C6701).

previous works in reducing the complexity of computation. Implementation in digital signal processor (DSP) for real-time digital watermarking becomes feasible.

The process of embedding watermark in digital image for copyright protection and marketing applications have been proposed over the past decade. Conventional technique is to embed a secret bit string in spatial, frequency, or wavelet domain into an image. The FDCT algorithm is implemented in a digital signal processor (TMS320C6701) to validate its efficiency in digital watermarking applications. The signal processor in both fixed-point and floating-point is supported by a set of software development tools with C/C++ compiler, assembly optimizer, a linker, and assorted utilities as shown in Figure 4. The proposed FDCT algorithm is written in C language, and the C/C++ compiler is able to perform optimization ($n = 0, 1, 2, 3$) in different level of clock cycles and code size. The lowest level ($n = 0$) optimization provides the operations of performing loop rotation, allocating variables to registers, and simplifying expressions and statements, the first level ($n = 1$) on constant propagation and unused assignments, the second level ($n = 2$) on software pipelining, unused global assignments, loop unrolling, and incremented pointer, and the highest level ($n = 3$) on optimization by simplifying functions with return values never used, removing all functions never called, inline calls to small functions, and reorder function declarations so that the attributes of called functions are known when the caller is optimized. $n = 0$ and 1 levels can efficiently reduce the code size, while $n = 2$ and 3 enhance the execution speed with larger code size. The computation time and code size in different levels of optimization are listed in Table 2. With increasing code size, the clock cycles and processing time decrease, so the best optimization is by $n = 3$.

Table 2: Processing time of the FDCT algorithm in DSP.

Optimization level (n)	Clocks (cycle)	Time (ms)	Code size (KB)
None	22,141,816	133	21
0	15,304,558	92	19
1	11,207,925	67	22
2	5,715,198	35	36
3	5,653,914	34	35
Matlab	NA	610	NA

For a digital watermark (32×32 block size) embedded in an original image (256×256), calculation by the FDCT

algorithm in frequency domain and then inverse transform of the encrypted data back to spatial domain image uses 5,653,914 clocks ($n = 3$), corresponding to 34 ms in 35 KB code size. Implementation shows that it takes only 0.24 seconds to have the watermark embedded in the original image. Extraction of watermark by inverse transform IDCT is within 0.21 seconds. Real-time implementation of the FDCT algorithm in DSP for image processing is shown very efficient.

5. Conclusions

(1) A fast discrete cosine transform (FDCT) algorithm that utilizes the energy compactness and matrix sparseness properties in frequency domain for higher computation performance is developed. For a JPEG image of 8×8 block size in spatial domain, the algorithm first decomposes the 2D DCT into one pair of 1D DCTs, and the calculation can be completed in only 24 multiplications. The 2D spatial data is a linear combination of base image obtained by the outer product of the column and row vectors of cosine functions such that the inverse DCT is as efficient. The algorithm is shown to achieve high performance compared to many other previous works.

(2) The algorithm optimizes a 2D DCT by exploiting the redundancy of the frequency coefficients so as to facilitate the implementation in digital signal processor (DSP). For a spatial domain data matrix $\mathbf{S}$, the 2D DCT data matrix $\mathbf{F}$ includes only a few nonzero elements in the low frequency range, which makes it possible to design efficient IDCT algorithm. The energy compactness property of $\mathbf{F}$ and its inverse matrix $\mathbf{S}'$ can be written as linear combinations of the cosine functions such that both FDCT and its inverse transform are shown to have the same coefficients and matrix blocks for efficient hardware implementation.

(3) An example of digital image watermarking is applied to demonstrate the efficiency of the FDCT algorithm. Hardware implementation of watermarking in DSP shows that it takes only 0.24 seconds to embed a 32×32 block size digital watermark into a digital image of block size 256×256. Implementation also shows that extraction of watermark can be completed within 0.21 seconds. The FDCT algorithm in DSP is shown efficient and effective in real-time implementation of digital image watermarking.

Competing Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *Institute of Electrical and Electronics Engineers. Transactions on Computers*, vol. 23, pp. 90–93, 1974.

[2] C. Sun and E.-H. Yang, "An efficient DCT-based image compression system based on Laplacian transparent composite model," *IEEE Transactions on Image Processing*, vol. 24, no. 3, pp. 886–900, 2015.

[3] M. Jridi, A. Alfalou, and P. K. Meher, "Optimized architecture using a novel subexpression elimination on Loeffler algorithm for DCT-based image compression," *VLSI Design*, vol. 2012, Article ID 209208, 12 pages, 2012.

[4] P. R. Kumbhare and U. M. Gokhale, "Design and implementation of 2D-DCT by using Arai algorithm for image compression," *Journal of The International Association of Advanced Technology and Science*, vol. 16, no. 5, article 5, 2015.

[5] M. Jridi, Y. Ouerhani, and A. Alfalou, "Low complexity DCT engine for image and video compression," *Real-Time Image and Video Processing*, vol. 8656, pp. 1–9, 2013.

[6] M. Marimuthu, R. Muthaiah, and P. Swaminathan, "Sub-band based DCT for image compression," *Research Journal of Applied Sciences, Engineering and Technology*, vol. 4, no. 24, pp. 5387–5390, 2012.

[7] L.-T. Ko, J.-E. Chen, H.-C. Hsin, Y.-S. Shieh, and T.-Y. Sung, "A unified algorithm for subband-based discrete cosine transform," *Mathematical Problems in Engineering*, vol. 2012, Article ID 912194, 31 pages, 2012.

[8] T.-Y. Sung, Y.-S. Shieh, and H.-C. Hsin, "An efficient VLSI linear array for DCT/IDCT using subband decomposition algorithm," *Mathematical Problems in Engineering*, vol. 2010, Article ID 185398, 21 pages, 2010.

[9] M. Manoria and P. Dixit, "An efficient DCT compression technique using Strassen's matrix multiplication algorithm," *International Journal of Computer Applications*, vol. 60, no. 9, pp. 45–50, 2012.

[10] S. Khan, E. Casseau, and D. Menard, "High performance discrete cosine transform operator using multimedia oriented subword parallelism," *Advances in Computer Engineering*, vol. 2015, Article ID 405856, 10 pages, 2015.

[11] P. Agrawal and S. K. Sharma, "Review paper on image compression using DCT, KLT and DWT," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 4, no. 9, pp. 928–931, 2014.