A FAST 8×8 IDCT ALGORITHM FOR HEVC

Tianlong Ma, Cong Liu, Yibo Fan*, Xiaoyang Zeng
State Key Lab of ASIC & System, Fudan University, Shanghai 200433, China
* Email:fanyibo@fudan.edu.cn

Abstract

As an approximation to the Discrete Cosine Transform (DCT), Integer Cosine Transform (ICT) is widely used in latest video coding standards, such as H.264/AVC, VC-1 and AVS. High Efficiency Video Coding (HEVC), the next generation of video compression standard, adopts 4/8/16/32 integer transform. Since the size of matrices themselves and the numerical magnitude of matrix elements are very large, HEVC transform suffers from huge computational complexity. To alleviate this problem, we proposed a fast algorithm for order-8 integer transform for HEVC. This algorithm has 66% less multiplications and 46% less additions than direct method and saves 60% area for hardware implementation. It is illustrated by signal-flow graph, which is easy to be translated to hardware or software implementation.

1. Introduction

The Discrete Cosine Transform (DCT) [1] has been widely applied in the area of image compression and video compression, such as JPEG, MPEG-2/4 and H.263. Its popularity is attributed to its ability to decorrelate data of spatial domain into data of frequency domain. Data will become more compact after being transformed, redundant information can be further removed. The DCT is considered as the closest to K-L transform, which is the ideal energy compaction transform. However, the matrix elements of DCT contain real numbers presented by a finite number of bits, which inevitably leads to the possibility of drift (mismatch between the decoded data in the encoder and decoder). Several methods have been introduced to control the accumulation of drift in video compress standards before H.264. However, H.264 makes extensive use of prediction, which causes it to be very sensitive to drift [2]. In order to eliminate mismatch between encoders and decoders and to facilitate low complexity implementations, latest video standards like H.264, VC-1 and AVS begin to adopt integer transform.

High Efficiency Video Coding (HEVC) [3] is the newest standard for high definition video processing. It is considered as the successor of H.264. Main goal of HEVC is to achieve 50% higher coding efficiency than H.264. In order to achieve this goal, HEVC adopts lots of state of the art coding tools including 4/8/16/32 integer transform. Compared to H.264, not only the size of matrices themselves but also matrix elements get larger. This makes the implementation of both hardware and software become very complicated. In this work, a fast algorithm for 8x8 integer transform of HEVC is presented, which is suitable for hardware and software implementation.

The rest of the paper is organized as follows. Section 2 describes the details of the fast algorithm. Section 3 gives some comparison. Finally, a conclusion is drawn in Section 4.

2. Proposed fast 8x8 IDCT algorithm

The two dimensional inverse DCT is

$$f = A^T \times F \times A$$

It can be implemented using two 1-D IDCT, with a transpose memory as its intermediate memory. In this architecture, it adopts the row-column decomposition approach. It performs 1-D operation on each row followed by another 1-D operation on each column, as Figure 1 shows.

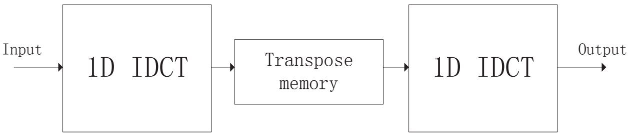

Figure 1. 2D IDCT architecture with transpose memory

The Integer Cosine Transform (ICT) is an approximation to discrete cosine transform [4]. Generally, integer cosine transform must meet some restrictions. Details will not be covered here. Interested readers may refer to Reference [4]. Integer cosine transform inherits symmetric property from DCT between the left side and right side of the matrix. That is, in 2^M (2^M=N, M>1) order integer cosine transform matrix, for any row $[a_{i0}, a_{i1}, \ldots, a_{i(N-1)}]$, it has properties as follows

$$a_{ix} = a_{i[N-1-x]}, i = 0, 2, 4, \cdots, N-2$$

$$a_{ix} = -a_{i[N-1-x]}, i = 1, 3, 5, \cdots, N-1$$

Where

$$x = 0, 1, 2, \cdots, N-1$$

According to this property [5] [6], we can decompose a 2^M (2^M=N, M>1)order integer cosine transform matrix as follows

978-1-4673-6417-1/13/$31.00 ©2013 IEEE

$$A_N = P_N \times \begin{bmatrix} A_{N/2} & 0 \\ 0 & R_{N/2} \end{bmatrix} \times B_N \qquad (1)$$

P_N is permutation matrix

$$\begin{aligned} P_N(i,j) &= 1 \qquad \text{for } i=2*j \text{ or } i=(j-N/2)*2+1 \\ &= 0 \qquad \text{otherwise} \end{aligned}$$

B_N is butterfly operation matrix

$$B_N = \begin{bmatrix} I_{N/2} & \hat{I}_{N/2} \\ \hat{I}_{N/2} & -I_{N/2} \end{bmatrix}$$

Where

$I_{N/2}$ is an identity matrix of order N/2, $\hat{I}_{N/2}$ is the opposite diagonal identity matrix.

$A_{N/2}$ is the even part which comes from the even rows of A_N. $A_{N/2}$ is also the order N/2 integer cosine transform matrix and can continue to decompose in the same way. $R_{N/2}$ is the odd part which comes from the odd rows of A_N. Reference [5] proposed a generalized method of decomposing the odd part for DCT. However, it can't be applied in integer transform.

HEVC specifies the 4x4, 8x8, 16x16 and 32x32 inverse transform. 4x4 matrix can be factorized through equation (1). 4x4 matrix of HEVC is defined as

$$A_4 = \begin{bmatrix} 64 & 64 & 64 & 64 \\ 83 & 36 & -36 & -83 \\ 64 & -64 & -64 & 64 \\ 36 & -83 & 83 & -36 \end{bmatrix}$$

We factorize A_4 as follows

$$A_4 = P_4 \times \begin{bmatrix} A_2 & 0 \\ 0 & R_2 \end{bmatrix} \times B_4 \qquad (2)$$

$$P_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad B_4 = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & -1 & 0 \\ 1 & 0 & 0 & -1 \end{bmatrix}$$

$$A_2 = \begin{bmatrix} 64 & 64 \\ 64 & -64 \end{bmatrix} \qquad R_2 = \begin{bmatrix} 36 & 83 \\ -83 & 36 \end{bmatrix}$$

For the 1-D IDCT transform

$$[m_0, m_1, m_2, m_3] = [f_0, f_1, f_2, f_3] \times A_4 \qquad (3)$$

$[f_0, f_1, f_2, f_3]$ is the input vector. $[m_0, m_1, m_2, m_3]$ is the output vector. Substituting equation (2) into equation (3), we can get the signal flow graph in Figure 2.

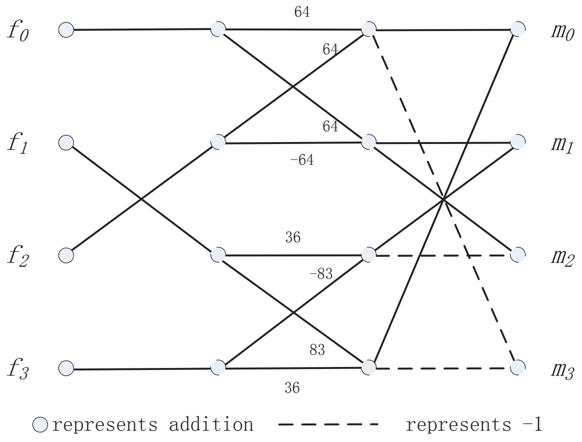

Figure 2. 4x4 1-D inverse DCT signal flow graph

8x8 matrix in HEVC is defined as

$$A_8 = \begin{bmatrix} 64 & 64 & 64 & 64 & 64 & 64 & 64 & 64 \\ 89 & 75 & 50 & 18 & -18 & -50 & -75 & -89 \\ 83 & 36 & -36 & -83 & -83 & -36 & 36 & 83 \\ 75 & -18 & -89 & -50 & 50 & 89 & 18 & 75 \\ 64 & -64 & -64 & 64 & 64 & -64 & -64 & 64 \\ 50 & -89 & 18 & 75 & -75 & -18 & 89 & -50 \\ 36 & -83 & 83 & -36 & -36 & 83 & -83 & 36 \\ 18 & -50 & 75 & -89 & 89 & -75 & 50 & -18 \end{bmatrix}$$

A_8 can be factorized by utilizing equation (1)

$$A_8 = P_8 \times \begin{bmatrix} A_4 & 0 \\ 0 & R_4 \end{bmatrix} \times B_8$$

$$P_8 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_4 = \begin{bmatrix} 18 & 50 & 75 & 89 \\ -50 & -89 & -18 & 75 \\ 75 & 18 & -89 & 50 \\ -89 & 75 & -50 & 18 \end{bmatrix}$$

$$B_8 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}$$

The decomposition of R_4 is the core idea of our proposal. First of all, we need to separate R_4 into the form of sum of two matrices.

$$R_4 = \mu + \lambda$$

Where

$$\lambda = \begin{bmatrix} 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$\mu = \begin{bmatrix} 18 & 50 & 75 & 90 \\ -50 & -90 & -18 & 75 \\ 75 & 18 & -90 & 50 \\ -90 & 75 & -50 & 18 \end{bmatrix}$$

And μ can be factorized as follows

$$\mu = a \times b \times c$$

Where

$$a = \begin{bmatrix} 1 & 0 & 0 & 5 \\ 0 & -4 & 6 & 0 \\ 0 & 6 & 4 & 0 \\ -5 & 0 & 0 & 1 \end{bmatrix}$$

$$b = \begin{bmatrix} 0 & 0 & 25/2 & 18 \\ 0 & 0 & -9 & 25/2 \\ 25/2 & -9 & 0 & 0 \\ 18 & 25/2 & 0 & 0 \end{bmatrix}$$

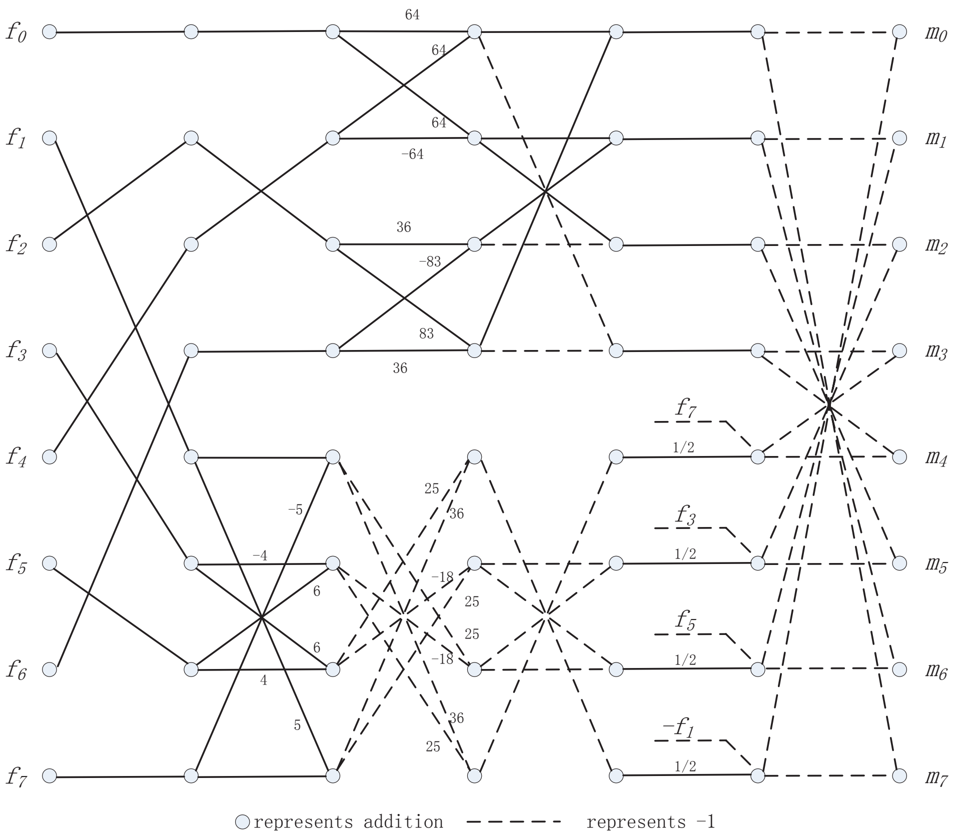

Figure 3. 8x8 1-D inverse DCT signal flow graph for HEVC

$$c = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & -1 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

a, b, c and λ are all sparse matrix, which make the 8x8 transform suitable for implementation.

1-D 8x8 inverse DCT is as follows

$$[m_0, m_1, m_2, m_3, m_4, m_5, m_6, m_7] =$$
$$[f_0, f_1, f_2, f_3, f_4, f_5, f_6, f_7] \times A_8$$

In order to avoid the loss of information caused by non-integer elements of matrix b, we decompose R_4 like this:

$$R_4 = \mu + \lambda = [a \times (2b) \times c] / 2 + \lambda$$

The factorization of A_4 has been shown above, combined with the decomposition of R_4, we can get the signal flow graph of 8x8 inverse DCT of HEVC in Figure 3.

3. Comparison

In this section, we give complexity comparison with two other algorithms, which is listed in Table 1.

Table 1. Complexity comparison

	Multipli-cations	Additions	Hardware area(um^2)
Algorithm A	64	56	105291
Algorithm B	24	28	49507
Algorithm C	22	30	41520

Algorithm A represents the direct 8x8 transform without any fast algorithm. Algorithm B represents the conventional fast algorithm similar to C and without R_4 decomposition, which is adopted in [6]. Algorithm C represents the fast algorithm proposed in this work.

We can see from Table 1 that the proposed algorithm has 66% less multiplications and 46% less additions than direct method. It clearly shows the superiority of the proposed algorithm over the original direct algorithm in terms of computational complexity. Algorithm C has two more additions but two less multiplications than algorithm B. And the multiplication factor of algorithm C is smaller than algorithm B. Since multiplication is much more area-cost than addition in hardware implementation, algorithm C can save expenses for hardware implementation.

The 1D IDCT of three algorithms are implemented in Verilog HDL and synthesized by Synopsys Design Compiler using SMIC 130nm CMOS library. They are all combinational circuits. The hardware areas are listed in Table 1. The area of algorithm A is about two times than B and C, which is the same as the ratio of additions and multiplications. Area of algorithm B is a little larger than algorithm C, which is also as expected.

4. Conclusion

In this work, a fast 8x8 IDCT algorithm for HEVC is proposed. It utilizes the symmetric property of integer cosine transform to divide matrix into even part and odd part. As for the 8x8 transform matrix in HEVC, the even part and odd part can continue to decompose into sparse matrices. This algorithm has 66% less multiplications and 46% less additions than direct method and saves about 60% area for hardware implementation.

Acknowledgement

This paper is supported by National High Technology Research and Development Program (863, 2012AA012001), State Key Lab of ASIC & System Project (11MS004), Specialized Research Fund for the Doctoral Program of Higher Education (SRFDP, 20120071120021).

References

[1] N. Ahmed, T. Natarajan and K.R. Rao, "Discrete Cosine Transform," IEEE Transactions on Computers, vol.C-23, no.1, pp.90-93, Jan 1974.

[2] H.S. Malvar, A. Hallapuro, M. Karczewicz and L. Kerofsky, "Low-complexity transform and quantization in H.264/AVC," IEEE Transactions on Circuits and Systems for Video Technology, vol.13, no.7, pp. 598- 603, July 2003.

[3] Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, "High Efficiency Video Coding (HEVC) text specification draft 10. ", JCTVC-L1003.doc (2013).

[4] W.-K. Cham, "Development of integer cosine trans-forms by the principle of dyadic symmetry," IEE Proceedings I, Communications, Speech and Vision, vol.136, no.4, pp. 276- 282, Aug 1989.

[5] Wen-Hsiung Chen, C. Smith and S. Fralick, "A Fast Computational Algorithm for the Discrete Cosine Transform," IEEE Transactions on Communications, vol.25, no.9, pp. 1004- 1009, Sep 1977.

[6] Sha Shen, Weiwei Shen, Yibo Fan and Xiaoyang Zeng, "A Unified 4/8/16/32-Point Integer IDCT Architecture for Multiple Video Coding Standards," 2012 IEEE International Conference on Multimedia and Expo (ICME), vol., no., pp.788-793, 9-13 July 2012.