

Integer DCT-based Image Coding

Changjiang Wei Pengwei Hao Qingyun Shi
National Lab on Machine Perception, Peking University
Beijing, 100871, CHINA

Abstract

DCT-based image/video coding is still popular now. In this paper, a novel embedded image coding scheme based on integer reversible DCT is proposed. It integrates lossy and lossless coding schemes perfectly. The transform is implemented by factoring the float DCT transform matrix into a series of integer reversible transform matrices. We apply the series of matrices to image samples, and encode the coefficients by several effective coding algorithms. Our simulation results show that the integer DCT coding method is superior to the float DCT coding method for lossless coding, and the performances are close for lossy coding.

1. Introduction

Image coding is of great interest in applications where efficiency with respect to data storage or transmission bandwidth is sought. Transform based image coding has become the main methods of image coding and video coding, such as JPEG, JPEG2000, MPEG-1, MPEG-2, H.263, MPEG-4 etc.

The discrete cosine transform (DCT) has been applied extensively to the area of image coding. It has nice decorrelation and excellent energy-compaction properties, and as a result it was chosen as the basis for the Joint Photography Experts' Group (JPEG) still-picture compression standard. Moreover, DCT can be easily implemented by hardware. Nevertheless these methods adopted float DCT which is not integer reversible. If we have to use float DCT to implement lossless coding, the corresponding residual image must be encoded which would result in the decrease of coding efficiency. In [4], the author presented a DCT-based lossless coding scheme, the quantization is only for the high-energy DCT coefficients in each block, an inverse DCT of only these quantized coefficients are carried out, then an error residual sequence is formed to be coded. The number of

coefficients used in this scheme is determined by using a performance metric for compression. Furthermore, a simple differencing scheme is performed on the coefficient that exploits correlation between high-energy DCT coefficients in neighboring blocks of an image. Compared with ours, the efficiency of their method is low. The essential feature of our method is that our transformation is integer reversible, so lossless coding can be implemented by encoding the transform coefficients losslessly.

As we know, many effective still image coding schemes are DWT-based, such as EZW^[5], SPIHT^[6], EBCOT^[7], etc. The high performances of these DWT-based algorithms are partly owing to the wavelet transform, but we must emphasize that much of the performance gain is obtained by carefully designed quantizers (e.g., zerotree quantizer) that are tailored to transform structure. All of these adopted the effective bit-plane quantization techniques and took full advantage of the regularities of distributions of the transform coefficients that are not particular features of wavelets. So, for other transforms we can also utilize the coding algorithm. Moreover DCT has many advantages over DWT, especially in video coding, the ongoing video coding specifications are DCT-based, so the DCT presented by our paper can be applied to video coding. Our method integrates integer DCT with many coding schemes, such as EZW, SPIHT, EBCOT that preserve both the finer feature of DCT and good coding efficiency.

In Section II we will introduce the integer DCT; the encoding pass is discussed in Section III, and some experimental results are illustrated in Section IV.

2. Integer DCT

In [3], Yan proposed an integer DCT method by factoring the butterfly calculations of DCT. In [1],

Hao presented the general algorithm of reversible linear transforms, the conclusion is, a linear transform has integer implementations as long as it is invertible and finite dimensional. In paper [2], Hao obtained optimal results, (i) A linear transform has integer implementations as long as it is invertible and finite dimensional. (ii) For a N-by-N nonsingular matrix, there are not more than 3 TERMS or not more than N+1 SERMs except a possible permutation matrix in the TERM or the SERM factorization. In this paper, we use the factorization method to obtain the series of integer DCT transform matrices.

2.1 Matrix factorization

For 8 by 8 DCT transform matrix A , with the method of [2], we can obtain many factorization results. Through optimized selection, we choose the following factorization:

$$A = P_L S_8 S_7 S_6 S_5 S_4 S_3 S_2 S_1 S_0 P_R$$

$$S_m = I + e_m s_m^T, m = 1, 2, \dots, 8, S_0 = I + e_8 s_0^T$$

Where, e_m is the m -th column vector of 8-order identity matrix, s_m is a vector that the m -th element is 0, where $m = 1, 2, \dots, 8$, s_0 is a vector that the 8th element is 0.

In following text, we will show the transform of these matrices are integer reversible.

Consider transform $y = S_m x$, x presents integer signal, from following equation

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_m \\ \vdots \\ y_N \end{bmatrix} = S_m x = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 \\ \cdots & \ddots & \cdots & \cdots & \cdots \\ s_{m,1} & \cdots & 1 & \cdots & s_{m,N} \\ \cdots & \cdots & \cdots & \ddots & \cdots \\ 0 & \cdots & \cdots & \cdots & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_m \\ \vdots \\ x_N \end{bmatrix}$$

We can get the integer transform result:

$$y_k = \begin{cases} x_k, & k \neq m \\ x_m + \left[\sum_{i=1}^{m-1} s_{m,i} x_i + \sum_{i=m+1}^N s_{m,i} x_i \right], & k = m \end{cases}$$

Where $k = 1, 2, \dots, N$, $[\]$ denotes integer conversion, can be rounding, chopping, carry-in, or some other

styles.

Obviously, reverse transform is

$$x_k = \begin{cases} y_k, & k \neq m \\ y_m - \left[\sum_{i=1}^{m-1} s_{m,i} y_i + \sum_{i=m+1}^N s_{m,i} y_i \right], & k = m \end{cases}$$

Then we can see that the linear transform $y = S_m x$ can be implemented reversibly for integers, so the transform $y = Ax$ can also be implemented for integers step by step. The general transform procedures are: first, to permute the input signal if needed, then to apply $y = S_m x$ ($m = 0, 1, 2, \dots, 8$) in turn for elementary integer transform steps, and finally to permute the output signal y if needed. So y is the integer result of transform and reversible. It is an approximated implementation of the original float DCT, but it is integer reversible.

There are no selection algorithms given in [2]. So, we need select the optimal factorization result from matrix factorizations. Here we present a nearly-optimal selection method. We define optimal rule to make integer transform results close to float ones. Consider integer transform $[y] = [Ax]$, let

d_k be the error caused by integer conversion in S_k transform calculation, we have,

$$\begin{aligned} [y] = [Ax] &= P_L (S_8 (\cdots (S_1 (S_0 P_R x + d_0) + d_1) + \cdots + d_7) + d_8) \\ &= Ax + P_L \left(\sum_{k=0}^7 \prod_{m=k+1}^8 S_m d_k + d_8 \right) = Ax + P_L \left(\sum_{k=0}^7 T_k d_k + d_8 \right) \end{aligned}$$

where $T_k = \prod_{m=k+1}^8 S_m$, $k = 0, 1, 2, \dots, N-1$.

For the form of S_k , the elements of d_k are all zeros except the k -th one, for $k = 1, 2, \dots, 7$. The elements of d_0 are all zeros except the 8th one. So we can select the factorization result which makes the absolute value of the k -th column of all T_k and the 8th column of T_0 as small as possible, where $k = 1, 2, \dots, 7$. We adopt a local search method to select the nearly-optimal factorization result. Finally,

we choose following factorization result:

$$\begin{aligned}
s_0 &= [1.165, 1.236, 1.201, 1.014, -0.367, 0.442, -1.962, 0]^T \\
s_1 &= [0, 1.033, 0.364, 0.007, -0.361, 0.395, -0.715, -0.462]^T \\
s_2 &= [-0.377, 0, 0.532, 0.199, -0.450, 0.608, -0.876, -0.272]^T \\
s_3 &= [0.424, -0.836, 0, 0.721, -0.701, 0.436, -0.847, -0.163]^T \\
s_4 &= [0.589, -0.160, 0.027, 0, 0.414, 0.329, -0.898, -0.227]^T \\
s_5 &= [0.067, 0.560, 0.759, -0.537, 0, 0.324, -0.130, -0.320]^T \\
s_6 &= [-0.342, 0.216, 0.270, -0.191, -1.082, 0, 0.293, -0.347]^T \\
s_7 &= [-0.058, -0.306, -0.382, 0.270, 0.531, 0.108, 0, 0.490]^T \\
s_8 &= [0.272, -1.471, -0.978, -0.016, 1.803, 1.162, -1.318, 0]^T
\end{aligned}$$

The corresponding left (row) and right (column) permutation matrix P_L and P_R are:

$$P_L = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad P_R = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

2.2 Rearrangement of transform coefficients

Having transformed an image, we can encode the transform coefficients with some wavelet-based coding schemes, such as SPIHT and EBCOT. In order to apply these coding schemes we rearrange the coefficients to take full advantage of these schemes. We adopt two methods:

First, we regard each sample in an 8x8 block as one frequency coefficient, and then collect the coefficients of the same frequency from all the blocks according to the block order (figure 2). For transform coefficients matrix $D[M][N]$, the matrix of coefficients rearranged became $W[M][N]$. There is a relation between them: $W[(i\%8)M/8 + i/8][(j\%8)N/8 + j/8] = D[i][j]$, where $i = 1, 2, \dots, M, j = 1, 2, \dots, N$, % denotes modulus operator. This rearranging method corresponds to the complete decomposition of wavelet coefficients. A complete decomposition is that all the four subbands are decomposed into four more subbands as well. As a result, the sizes of all

subbands are the same.

Second, we classify each 8x8 block to 10 different frequency subbands (figure3), and then we rearrange the coefficients of same frequency in each block according to the block order. This rearranging method corresponds to the tree decomposition of wavelet coefficients.

3 Encoding

3.1 Encoding algorithms with complete decomposition

In [7], David Taubman proposed EBCOT (Embedded Block Coding with Optimized Truncation) algorithm, which became the coding scheme kernel of JPEG 2000, the new generation image compression standard. The algorithm supports many wavelet decomposition forms, partitions the subbands into a collection of relatively small code-blocks after wavelet decomposition, and encodes each code-block nearly independently. One of its advantages is that it is independent to decomposition forms. We apply the first coefficient rearrangement method in Section 1.2 to encode the integer DCT coefficients by EBCOT. The simulation results are shown in Section 4.

3.2 Encoding algorithms with tree decomposition

In [5], Shapiro presented the well-known EZW algorithm. An then, Said and Pearlman presented the more effective SPIHT algorithm. The main contribution of EZW is zerotree quantization of wavelet coefficients, which works by efficiently predicting the children nodes based on significance/insignificance of their parent. An embedded zerotree quantizer refines each input coefficient sequentially using a bitmap-like of coding scheme, and it stops when the size of encoded stream reaches the exact target bit rate. Said and Pearlman described an SPIHT coder that achieves about 1 dB gain in PSNR over ZEW at the same bit rate for typical images. Here we use the second coefficients rearrangement method in Section 1.2. We encode the integer DCT coefficients by SPIHT whose performance is higher than that by EZW. The simulation results is presented in next section.

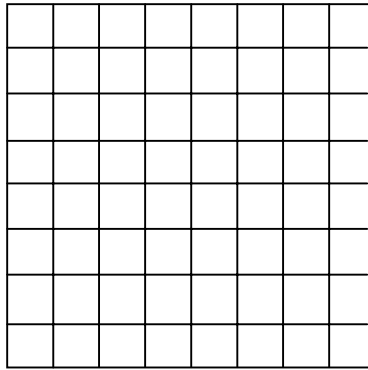


Figure 2 8x8 DCT coefficients (complete decomposition)

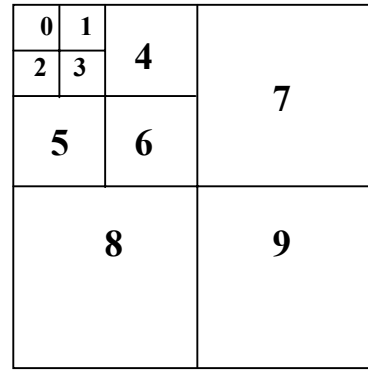


Figure 3 3 level subband form (tree decomposition)

4 Experimental results

We use several commonly-used gray-level images to test our image coding schemes. The images are Lena, Peppers and Mandrill, which have the same size of 512x512x8bits, 262144 bytes. We compute the entropy of the transformed data to assess the lossless coding results and the PSNR to evaluate lossy coding experiments. In table I, we can find, since the different information encoded, the lossless coding with float DCT performs worse than that with our integer DCT, which is the very advantage of integer DCT. We also tested the performance of lossy coding with float DCT and integer DCT. It follows our expectation that their coding results are very similar, since the entropy of integer DCT is close to that of rounded float DCT.

We also tested the lossy coding performance with the two DCT implementations. The results are in table II. We can see from the table, for lossy coding, their coding results are very close, it is because the entropy of integer DCT is close to that of rounded float DCT. The same results and conclusion can be got from the tests with Peppers and Mandrill.

Table I Lossless compression results - entropy

Image	Lena	Peppers	Mandrill
Algorithm by [4]	5.02	5.46	7.41
Integer 8x8 DCT with SPIHT	4.35	4.63	6.06
Integer 8x8 DCT with EBCOT	4.49	4.80	6.24
Integer 16x16 DCT with SPIHT	4.29	4.62	6.00
Integer 16x16 DCT with EBCOT	4.49	4.80	6.24

Table II Lossy compression results (PSNR for image Lena)

Compression ratio	JPEG with float DCT	JPEG with integer DCT
2.8	43.62	43.16
12.5	35.78	35.72
32.5	30.40	30.25

5. References

- [1] Pengwei Hao, Qingyun Shi, "the integer implementation of reversible linear transform". China Science (Serie E), April 2000, No. 2, pp. 132-141
- [2] Pengwei Hao, Qingyun Shi, "Matrix Factorizations for Reversible Integer Mapping", accepted for publication to IEEE Transactions on Signal Processing.
- [3] Yusong Yan, Qingyun Shi, "reversible integer DCT and lossless image coding", Software Journal, Vol.11, No.5, pp.620-627, May, 2000
- [4] Giridhar Mandyam, Nasir Ahmed, and Neeraj Magotra, "Lossless Image Compression Using the Discrete Cosine Transform", Journal of Visual Communication and Image Representation, Vol.8, No.1, pp. 21-26, March, 1997
- [5] J.M.Shapiro, "Embedded image coding using zerotrees of wavelet coefficients". IEEE Trans. Signal Processing. Vol.41, pp. 3445-3463, Dec. 1993
- [6] A.Said and W.Pearlman, "A New, Fast and Efficient Image Codec based on Set Partitioning in Hierarchical Trees". IEEE Trans. Circuits and Systems for Video Technology. Vol. 6, no. 3, pp. 243-250, June 1996
- [7] David Taubman, "High Performance Scalable Image Compression With EBCOT". IEEE Trans. Image Processing. Vol. 9, no. 7, July 2000