Low Complexity Integer Transform and Adaptive Quantization Optimization

Si-Wei Ma\textsuperscript{1} (马思伟) and Wen Gao\textsuperscript{1,2} (高 文)
\textsuperscript{1}Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100080, P.R. China
\textsuperscript{2}Department of Computer Science, Harbin Institute of Technology, Harbin 150001, P.R. China
E-mail: {swma, wgaoo}@jdl.ac.cn
Received October 31, 2005; revised March 8, 2006.

Abstract In this paper, a new low complexity integer transform is proposed, which has been adopted by AVS1-P7. The proposed transform can enable AVS1-P7 to share the same quantization/dequantization table with AVS1-P2. As the bases of the proposed transform coefficients are very close, the transform normalization can be implemented only on the encoder side and the transform table size can be reduced compared with the transform used in H.264/MPEG-4 AVC. Along with the feature of the proposed transform, adaptive dead-zone quantization optimization for the proposed transform is studied. Experimental results show that the proposed integer transform has similar coding performance compared with the transform used in H.264/MPEG-4 AVC, and would gain near 0.1dB with the adaptive dead-zone quantization optimization.

Keywords AVS, discrete cosine transform (DCT), integer transform, quantization

1 Introduction

Transform coding is an important video coding technique and has been widely used in many international video coding standards, such as MPEG-1/2, H.261/2/3, etc. After transform, the spatial redundancy is attenuated and the compression can be achieved by the following quantization and entropy coding. From the energy compaction viewpoint, KLT (Karhunen-Loeve transform) is the best transform. However, it is difficult to use KLT in image and video coding because it is signal-dependent. DCT is a better approximation of KLT and easy for implementation due to its signal independence. But the float point multiplication in DCT is too complex and cannot map integer to integer losslessly.

In the past, many researches have been done on the integer-friendly approximation of the float DCT, such as binDCT\textsuperscript{[1,2]} and IntDCT\textsuperscript{[3]}, where the float DCT coefficient is approximated as an integer coefficient by a multiplier and a right shift. So the DCT transform can be implemented only by using shift and add. In the development of H.264, the transform is improved, and the integer transform was first proposed to H.264 in [4] (in fact, it was first used in TML). Then many techniques on integer cosine transform (ICT, or integer transform: IT) have been proposed to H.264, such as 16 × 16 ICT\textsuperscript{[5]}, 4 × 4 IT\textsuperscript{[6,7]} and adaptive block transform (ABT)\textsuperscript{[8]}. Finally, a low complexity 4 × 4 integer transform was adopted\textsuperscript{[9]} (8 × 8 transform was developed for H.264 the FRx profile later, and ABT was adopted by the FRx profile too\textsuperscript{[10,11]}). The integer transform has many advantages, such as low complexity, exact invertibility, etc. Combined with the normalization to the integer transform, a division-free quantization scheme is used in H.264/MPEG-4 AVC. But in the quantization/dequantization scheme of H.264/MPEG-4 AVC, a big quantization/dequantization table should be stored to reach transform normalization and dequantization/quantization, as the bases of the transform coefficients are different.

AVS is the latest video/audio coding standard developed in China. In AVS standard, the part 2 and the part 7 are two video coding standards, called as AVS1-P2 and AVS1-P7 respectively. In AVS, the transform was improved and a new kind of integer transform was developed\textsuperscript{[12,13]}. AVS transform is characteristic as the bases of the transform coefficients are very close, so that the transform normalization can be realized only on the encoder side, and that the quantization table size in the decoder is reduced. In this paper a low complexity integer transform is proposed, which has been adopted by AVS1-P7. In conjunction with the new characteristic of the proposed transform, adaptive dead-zone quantization optimization is also studied. Experimental results show that the proposed transform has similar coding performance as that of H.264/MPEG-4 AVC, and would gain nearly 0.1dB with the optimized adaptive dead-zone quantization.

The rest of the paper is organized as follows. Section 2 makes an in-depth study on the integer transform and quantization scheme in H.264/MPEG-4 AVC. In Section 3, the proposed transform and adaptive dead-zone quantization are discussed. Experimental results are provided in Section 4. Section 5 concludes the paper.

2 Integer Transform

A typical 4 × 4 orthogonal transform can be expressed as:

\[ T = \begin{bmatrix}
    a & b & c & d \\
    e & f & g & h \\
    i & j & k & l \\
    m & n & o & p
\end{bmatrix} \]
\[ A = \begin{bmatrix} a & a & a & a \\ b & c & -c & -b \\ a & -a & -a & a \\ c & -b & b & -c \end{bmatrix}. \]  

To keep low distortion from \(4 \times 4\) DCT, the following condition should be met:

\[ \frac{b}{c} \approx 2.4143 \]  

In H.264/MPEG-4 AVC, the forward transform is approximated with \(a = 1, b = 2, c = 1\); and the transform process can be expressed as:

\[ Y = AXA^T = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix} \begin{bmatrix} 1 & 1 & -1 & -2 \\ 2 & 1 & -1 & 2 \\ 1 & 1 & 1 & -2 \\ 1 & 2 & 1 & 1 \end{bmatrix}, \]

Since the integer transform is not a unitary matrix, \(Y\) must be normalized with:

\[ W = Y \odot E = \begin{bmatrix} y_{11} & y_{12} & y_{13} & y_{14} \\ y_{21} & y_{22} & y_{23} & y_{24} \\ y_{31} & y_{32} & y_{33} & y_{34} \\ y_{41} & y_{42} & y_{43} & y_{44} \end{bmatrix} \begin{bmatrix} 1/m^2 & 1/mm & 1/m^2 & 1/mm \\ 1/mm & 1/n^2 & 1/mm & 1/n^2 \\ 1/m^2 & 1/mm & 1/m^2 & 1/mm \\ 1/mm & 1/n^2 & 1/mm & 1/n^2 \end{bmatrix}, \]

where \(m = 2, n = \sqrt{10}\) are the norms of the first row and the second row of \(A\) respectively. In H.264/MPEG-4 AVC, the scale matrix \(E\) is combined together with quantization/dequantization process, in which the transform normalization and quantization can be reached only through the multiplication and right shift, as follows:

\[ Y_Q(i, j) = Y(i, j) \times E(i, j) / Q_{\text{step}}(QP) = Y(i, j) \times (QP, i, j) / 2^r, \]

where \(QP\) is the quantization parameter, and \(Q_{\text{step}}(QP)\) is the corresponding quantization step size. \(Q(PQ, i, j)\) is the quantization table on encoder side and \(Q(PQ, i, j)\) equals \(Q(PQ, i, j)\) in H.264/MPEG-4 AVC. For the inverse quantization, it is implemented in the same way as:

\[ Y_{DQ}(i, j) = Y_Q(i, j) \times E(i, j) \times Q_{\text{step}}(QP) = (Y_Q(i, j) \times DQ(QP, i, j)) / 2^r, \]

and

\[ Q_{\text{step}}(QP, i, j) \times M(i, j) \times DQ(QP, i, j) \times M(i, j) = 2^{r_1 + r_2}. \]

\(DQ(QP, i, j)\) is the dequantization table on the decoder side and \(DQ(QP, i, j)\) equals \(DQ(QP, i, j)\) in H.264/MPEG-4 AVC. The above transform, quantization, dequantization and inverse transform all can be implemented in 16-bit operation.

However, in H.264/MPEG-4 AVC, the quantization parameter \(QP\) ranges from 0 to 51. In hardware implementation, if we extend the quantization/dequantization table for all \(QP\), the table size would be \(52 \times 4 \times 4\). In fact, if all the rows of the transform have the same norm, the quantization/dequantization table size would be reduced to \(52 \times 1\). In old version of H.264/MPEG-4 AVC, such as TML9[15], an integer transform with \(a = 13, b = 17, c = 7\) in (1) was used. Every row of the transform has the same norm, and both the quantization and dequantization table size is \(32 \times 1\) (\(QP\) ranges from 0 to 31 for TML9). But the transform and quantization process in TML9 require 32-bit arithmetic operation that would increase hardware complexity.

In the development of AVS1-P2, an \(8 \times 8\) transform is adopted, shown as follows:


The transform has such a good feature that the bases of the transform coefficients are very close, so that the transform normalization can be realized only on the encoder side, and the dequantization table size on the decoder side is reduced. In the following section, the details for this feature will be discussed and a low complexity \(4 \times 4\) integer transform is proposed. Then adaptive quantization was studied for the new kind of integer transform. The proposed integer transform and quantization/dequantization can reduce the dequantization table size while using 16-bit arithmetic operation.

### 3 Low Complexity Integer Transform and Adaptive Quantization Optimization

#### 3.1 Low Complexity Integer Transform

To develop a low complexity \(4 \times 4\) transform from (1), besides the above constraint (2), an additional constraint should also be met:

\[ b^2 + c^2 = 2a^2. \]  

That means the basis function is the same, which would reduce transform normalization complexity. But only one solution exists for both (2) and (3) in the range \([0, 20]\), just as the transform used in TML9 (\(a = 13, b = 17, c = 7\)). In the development of AVS1-P7, we propose a new low complexity \(4 \times 4\) integer transform, where (2) and (3) are approximated with \(a = 2, b = 3, c = 1\), as the \(8 \times 8\) transform used in AVS1-P2. The proposed transform is shown as follows:

\[ T_A = \begin{bmatrix} 2 & 2 & 2 & 2 \\ 3 & 1 & -1 & -3 \\ 2 & -2 & -2 & 2 \\ 1 & -3 & 3 & -1 \end{bmatrix} \]