MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Efficient Macroblock Coding-Mode Decision for H.264/AVC Video Coding

Jun Xin, Anthony Vetro, Huifang Sun

TR2004-079 January 2004

Abstract

In this paper, we propose to use transform-domain processing in the macro block coding mode decision of H.264/AVC such that an optimal mode decision is achieved with significantly reduced computational complexity. Specifically, we achieve the computational savings by calculating the distortion and the residual error in the transform-domain. We show that the distortion calculation can be performed efficiently in the transform-domain such that the inverse transform and reconstruction of the pixels can be omitted. We calculate the residual error in the transform-domain by taking advantage of the fact that the transform of several intra prediction signals (DC, Horizontal, and Vertical) can be very efficiently calculated, and their transform coefficients have few non-zero entries.

Publication History:

1. First printing, TR-2004-79, 07 2004

Efficient Macroblock Coding-mode Decision for H.264/AVC Video Coding

Jun Xin, Anthony Vetro, and Huifang Sun
(jxin, avetro, hsun@merl.com)

Abstract

In this paper, we propose to use transform-domain processing in the macroblock coding-mode decision of H.264/AVC such that an optimal mode decision is achieved with significantly reduced computational complexity. Specifically, we achieve the computational savings by calculating the distortion and the residual error in the transform-domain. We show that the distortion calculation can be performed efficiently in the transform-domain such that the inverse transform and reconstruction of the pixels can be omitted. We calculate the residual error in the transform-domain by taking advantage of the fact that the transform of several intra prediction signals (DC, Horizontal, and Vertical) can be very efficiently calculated, and their transform coefficients have few non-zero entries.

I. Introduction

Major international video coding standards, including H.264/AVC [1], are based on a basic hybrid-coding framework that uses motion compensated prediction to remove temporal correlations and transforms to remove spatial correlations.

The basic encoding process of such a standard video encoder is shown in Figure 1. Each frame of an input video is divided into macroblocks. Each macroblock is subject to a transform/quantization, and entropy coding. The output of the transform/quantization is subject to an inverse quantization/transform. Motion estimation is performed, and a coding-mode decision is made considering the content of a pixel buffer. The coding-mode decision selects an optimal coding-mode. Then, the result of the prediction is subtracted from the input signal to produce an error signal. The result of the prediction is also added to the output of the inverse quantization/transform and stored into the pixel buffer.

The macroblock can be encoded as an intra-macroblock, which uses information from just the current frame. Alternatively, the macroblock can be encoded as an inter-macroblock, which is predicted using motion vectors that are estimated through motion estimation from the current and previous frames. There are various ways to perform intra-prediction and inter-prediction.

In general, each frame of video is divided into macroblocks, where each macroblock consists of a plurality of smaller-sized blocks. The macroblock is the basic unit of encoding, while the blocks typically correspond to the dimension of the transform. For instance, both MPEG-2 and H.264/AVC specify 16x16 macroblocks. However, the block size in MPEG-2 is

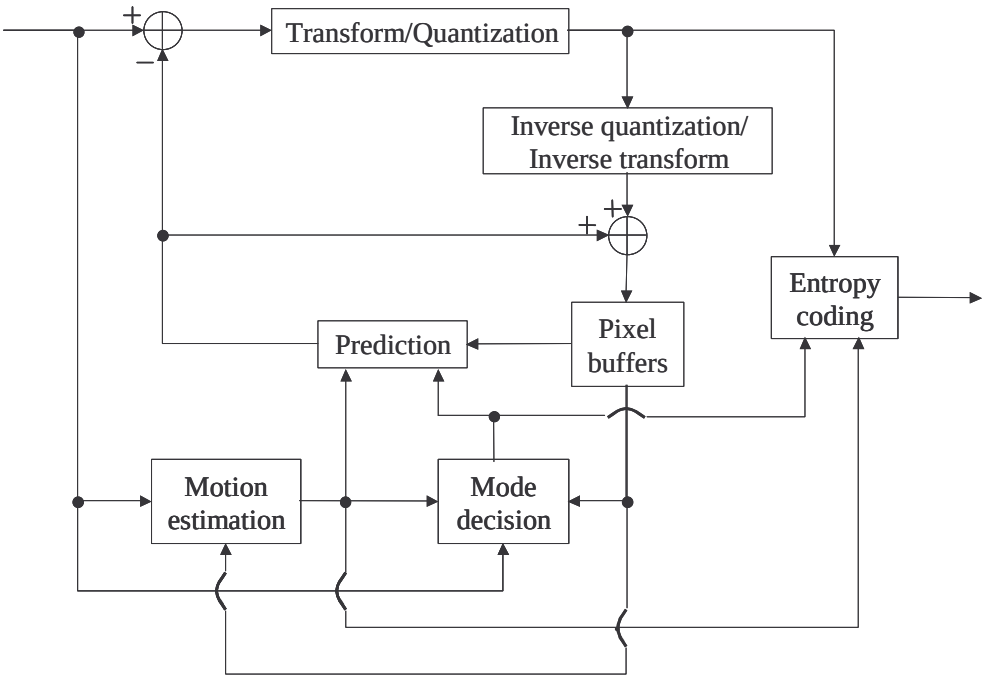

Figure 1. A standard video encoder based on hybrid DCT/MC.

8x8, corresponding to 8x8 DCT and Inverse DCT operations, while the block size in H.264/AVC is 4x4 corresponding to the H.264/AVC 4x4 transform (*HT*) and inverse transform operations.

We use the notion of *macroblock partition* to refer to the group of pixels in a macroblock that share a common prediction. The dimensions of a macroblock, block and macroblock partition are not necessarily equal. An allowable set of macroblock partitions typically vary from one coding scheme to another.

AVC defines a wide variety of allowable set of macroblock partitions. For instance, a 16x16 macroblock may have a mix of 8x8, 4x4, 4x8 and 8x4 macroblock partitions within a single macroblock. Prediction can then be performed independently for each macroblock partition, but the coding is still based on a 4x4 block.

The encoder selects the coding-modes for the macroblock, including the best macroblock partition and mode of prediction for each macroblock partition, such that the video coding performance is optimized. The selection process is conventionally referred to as *macroblock coding-mode decision*.

In H.264/AVC, there are many available modes for coding a macroblock. The available coding-modes for a macroblock in an I-slice include: *intra_4x4* prediction and *intra_16x16* prediction for luma samples, and *intra_8x8* prediction for chroma samples.

In the intra_4x4 prediction, each 4x4 macroblock partition can be coded using one of the nine prediction modes defined by the H.264/AVC standard. In the intra_16x16 and intra_8x8 predictions, each 16x16 or 8x8 macroblock partition can be coded using one of the four defined prediction modes. For a macroblock in a P-slice or B-slice, in additional to the coding-modes available for I-slices, many more coding-modes are available using various combinations of

macroblock partitions and reference frames. Every coding-mode provides a different rate-distortion (*RD*) trade-off.

Typically, the rate-distortion optimization uses a Lagrange multiplier to make the macroblock mode decision [2][3]. The rate-distortion optimization evaluates the Lagrange cost for each candidate coding-mode for a macroblock and selects the mode that yields a minimum Lagrange cost. The process for determining the Lagrange cost needs be performed many times because there are a large number of available modes for coding a macroblock according to the H.264/AVC standard. Therefore, the computation of the rate-distortion optimized coding-mode decision is very intensive. Consequently, there exists a need to perform efficient rate-distortion optimized macroblock mode decision in H.264/AVC video coding.

In this paper, we provide an efficient method for determining the Lagrange cost, which leads to an efficient, rate-distortion optimized macroblock mode decision. We will first give a brief review of the conventional rate-distortion optimized macroblock mode decision, and then present our proposed approach. We are currently working on simulations, and we will provide the results in later versions of the paper.

II. Rate-distortion Optimized Macroblock Mode Decision

If there are N candidate modes for coding a macroblock, then the Lagrange cost of the n^{th} candidate mode J_n, is the sum of the Lagrange cost of its associated macroblock partitions:

$$J_n = \sum_{i=1}^{P_n} J_{n,i} \qquad n = 1,2,...,N \tag{1}$$

Where P_n is the number of macroblock partitions of the n^{th} candidate mode. A macroblock partition can be of different size depending on the prediction mode. For example, the partition size is 4x4 for the intra_4x4 prediction, and 16x16 for the intra_16x16 prediction.

If the number of candidate coding-modes for the i^{th} partition of the n^{th} macroblock is $K_{n,i}$, then the cost of this macroblock partition is

$$\begin{aligned} J_{n,i} &= \min_{k=1,2,...,K_{n,i}} \left(J_{n,i,k} \right) \\ &= \min_{k=1,2,...,K_{n,i}} \left(D_{n,i,k} + \lambda \times R_{n,i,k} \right) \end{aligned} \tag{2}$$

Where R and D are respectively the rate and distortion, and λ is the Lagrange multiplier. The Lagrange multiplier controls the rate-distortion tradeoff of the macroblock coding, and may be derived from a quantization parameter. The above equation states that the Lagrange cost of the i^{th} partition of the n^{th} macroblock, $J_{n,i}$, is selected to be the minimum of the $K_{n,i}$ costs that are yielded by the candidate coding-modes for this partition. Therefore, the optimal coding-mode of this partition is the one that yields $J_{n,i}$.

The optimal coding-mode for the macroblock is selected to be the candidate mode that yields the minimum cost, i.e.,

$$J^* = \min_{n=1,2,...,N} J_n \tag{3}$$

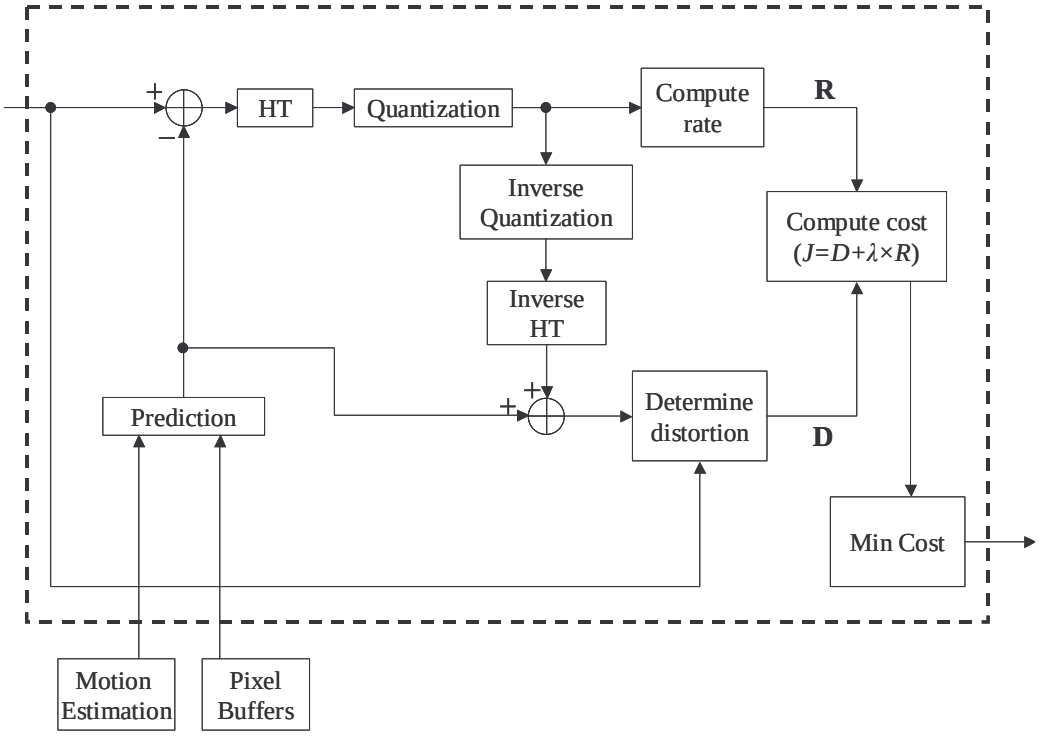

Figure 2. Conventional method of computing the cost for a coding mode.

Figure 2 shows the conventional method of computing the Lagrange cost for a coding-mode of a macroblock partition, i.e., $J_{n,i,k}$. The difference between the input macroblock partition and its prediction is HT-transformed, quantized, and then the rate is computed. The quantized HT-coefficients are also subject to inverse quantization (IQ), inverse HT-transform, and compensation to reconstruct the macroblock partition. The distortion is then computed between the reconstructed and the input macroblock partition. In the end, the Lagrange cost is computed using the rate and distortion. The optimal coding-mode then corresponds to the mode with the minimum cost.

III. The Proposed Mode Decision

Figure 3 shows the method and system, according to the paper, for selecting an optimal coding-mode, from multiple available candidate coding-modes, for each macroblock in a video.

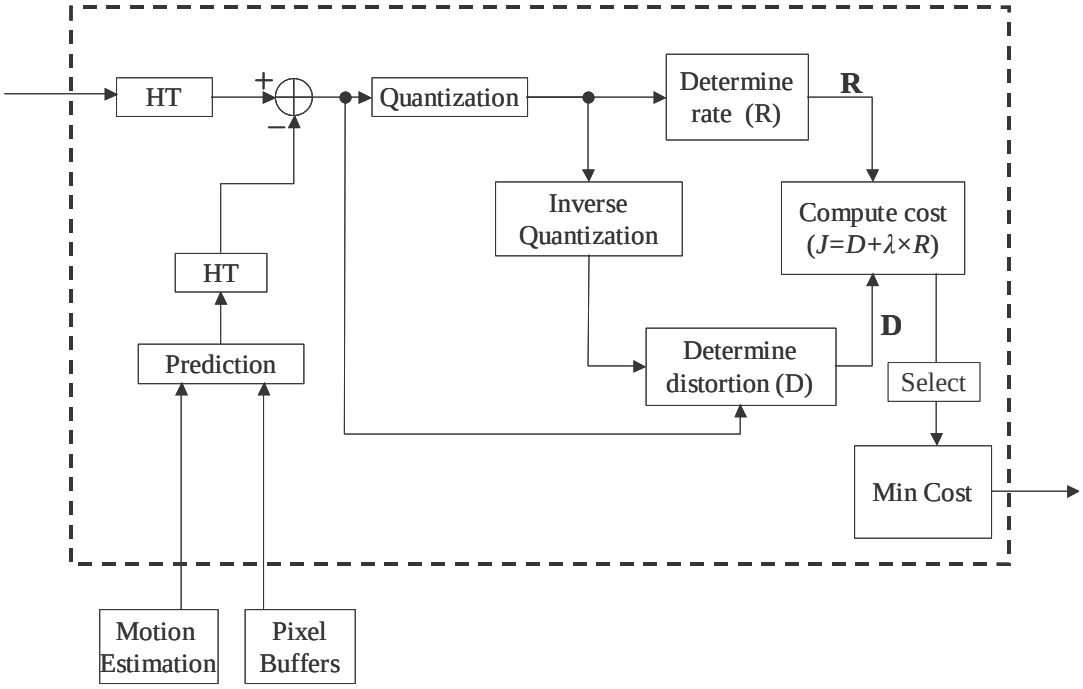

Figure 3. Proposed method of computing the cost for a coding mode.

Compare to the conventional method, shown in Figure 2, our invention has the following distinctive features.

1. The distortion is computed in the transform-domain instead of the pixel-domain, i.e., the distortion is computed directly using the HT-coefficients. Therefore, the inverse HT and the reconstruction of the macroblock partition are omitted. In the following, we will provide a method to calculate the distortion in the transform-domain such that it is approximately equal to the commonly used sum-of-squared-differences (SSD) in the pixel-domain.
2. The HT applies to both the input and the predicted macroblock partition, instead of the difference as in the conventional method. The HT of the input macroblock partition only needs to be performed once in the whole mode decision process, and the HT of the predicted partition needs to be performed for every prediction mode. Hence, our method needs to compute one more HT. However, as we describe below, the HT of the predicted signal can be very efficiently computed for some intra-prediction modes. The resulting savings (compared to computing the HT of the prediction error signal) more than offset the additional HT.

We have highlighted the use of the above method for efficiently performing the mode decision within the context of an encoding system. However, this method could also be applied to transcoding videos, including the case when the input and output video formats are based on different transformation kernels. In particular, when the above method is used in transcoding of intra-frames from MPEG-2 to H.264/AVC, the HT-coefficients of the input macroblock partition can be directly computed from the transform-coefficients of MPEG-2 video in the transform-domain [4]. In this case, the HT of the input macroblock partition is also omitted.

A) Determining Intra-Predicted HT-Coefficients

The conventional method for determining HT coefficients performs eight 1-D HT-transforms, i.e., four column-transforms followed by four row-transforms. However, certain properties of some intra-predicted signals can make the computation of their HT coefficients much more efficient.

In what follows, we describe efficient methods for determining the HT coefficients of the following intra-prediction modes: DC prediction, horizontal prediction, and vertical prediction. These prediction modes are used in the intra_4x4 and intra_16x16 predictions for luma samples, as well as the intra_8x8 prediction for chroma samples.

The following notations are used to in the details of the presentation.

p	– the predicted signal, 4x4 matrix
P	– HT-coefficients of p, 4x4 matrix
r, c	– row and column index, r,c=1,2,3,4
×	– multiplication
$(\bullet)^T$	– matrix transpose
$(\bullet)^{-1}$	– matrix inverse
H	– H.264/AVC transform (HT) kernel matrix, and

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}$$

In the DC prediction mode, the DC prediction value is dc, and we have

$$p_{dc}(r,c) = dc, \quad \text{for all } r \text{ and } c. \tag{4}$$

The HT of p_{dc}, P_{dc}, is all zero except the DC coefficient given by

$$P_{dc}(0,0) = 16 \times dc. \tag{5}$$

Therefore, only one operation is needed for the computation of the HT for DC prediction.

In the horizontal prediction mode, the prediction signal is denoted by

$$p_h = \begin{bmatrix} h1 & h1 & h1 & h1 \\ h2 & h2 & h2 & h2 \\ h3 & h3 & h3 & h3 \\ h4 & h4 & h4 & h4 \end{bmatrix} \tag{6}$$

Let $\mathbf{h} = [h1 \quad h2 \quad h3 \quad h4]^T$ be the 1D horizontal prediction vector. Then, the HT of p_h is

$$P_h = H \times \begin{bmatrix} h1 & h1 & h1 & h1 \\ h2 & h2 & h2 & h2 \\ h3 & h3 & h3 & h3 \\ h4 & h4 & h4 & h4 \end{bmatrix} \times H^T$$
$$= \begin{bmatrix} H \times \mathbf{h} & H \times \mathbf{h} & H \times \mathbf{h} & H \times \mathbf{h} \end{bmatrix} \times H^T \quad (7)$$
$$= \begin{bmatrix} 4 \times H \times \mathbf{h} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}$$

In the vertical prediction mode, the predicted signal is denoted by

$$p_v = \begin{bmatrix} v1 & v2 & v3 & v4 \\ v1 & v2 & v3 & v4 \\ v1 & v2 & v3 & v4 \\ v1 & v2 & v3 & v4 \end{bmatrix} \quad (8)$$

Let $\mathbf{v} = \begin{bmatrix} v1 & v2 & v3 & v4 \end{bmatrix}$ be the 1D vertical prediction vector. Then, the HT of p_v is

$$P_v = H \times \begin{bmatrix} v1 & v2 & v3 & v4 \\ v1 & v2 & v3 & v4 \\ v1 & v2 & v3 & v4 \\ v1 & v2 & v3 & v4 \end{bmatrix} \times H^T$$
$$= H \times \begin{bmatrix} \mathbf{v} \times H^T & \mathbf{v} \times H^T & \mathbf{v} \times H^T & \mathbf{v} \times H^T \end{bmatrix}^T \quad (9)$$
$$= \begin{bmatrix} 4 \times \mathbf{v} \times H^T & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}^T$$

Equations (7) and (9) suggest that the HT of horizontal/vertical prediction signals can be determined by a single 1-D transform, plus four shifting operations. This is much simpler than the eight 1-D transforms needed by the direct calculation.

For the above three prediction modes, the three predicted signals, P_{dc}, $P_{h,}$ and P_v, have mostly zero components: P_{dc} has just one non-zero component, P_h has non-zero values only in its first column, and P_v has non-zero values only in its first row. Therefore, the complexity of determining the difference between the input and the predicted HT-coefficients is also reduced.

Similar reductions in computation for the transformed prediction may also be possible for other modes, e.g., modes that predict along diagonal directions.

B) Determining Distortion in Transform-Domain

The SSD distortion in the pixel-domain is determined between the input signal and the reconstructed signal. The input signal, reconstructed signal, predicted signal, prediction error, and reconstructed prediction error are x, $\hat{x}$, p, e, $\hat{e}$, respectively. They are all 4x4 matrices. The SSD distortion D is

$$D = trace\left((x - \hat{x}) \times (x - \hat{x})^T\right)$$

Because $x = p + e$, and $\hat{x} = p + \hat{e}$,

$$D = trace\left((e-\hat{e})\times(e-\hat{e})^T\right) \tag{10}$$

If the HT of e is E, i.e., $E = H\times e\times H^T$, then, it follows that

$$e = H^{-1}\times E\times(H^T)^{-1} \tag{11}$$

The variable $\hat{E}$ is the signal whose inverse HT is $\hat{e}$, and taking into consideration the scaling after inverse HT in the H.264/AVC specification, we have

$$\hat{e} = \frac{1}{64}\left(\tilde{H}_{inv}\times\hat{E}\times\tilde{H}_{inv}{}^T\right), \tag{12}$$

Where $\tilde{H}_{inv}$ is the kernel matrix of the inverse HT used in the H.264/AVC standard, and

$$\tilde{H}_{inv} = \begin{bmatrix} 1 & 1 & 1 & 1/2 \\ 1 & 1/2 & -1 & -1 \\ 1 & -1/2 & -1 & 1 \\ 1 & -1 & 1 & -1/2 \end{bmatrix}$$

The goal is to determine D from E and $\hat{E}$, which are the input to the distortion computation block.

From (11) and (12), we have

$$\begin{aligned} e-\hat{e} &= H^{-1}\times E\times\left(H^T\right)^{-1} - \frac{1}{64}\left(\tilde{H}_{inv}\times\hat{E}\times\tilde{H}_{inv}{}^T\right) \\ &= \frac{1}{64}\left(H^{-1}\times 64\times E\times\left(H^T\right)^{-1} - \tilde{H}_{inv}\times\hat{E}\times\tilde{H}_{inv}{}^T\right) \end{aligned}$$

Let $M_1 = diag(4,5,4,5)$, and $\tilde{H}_{inv} = H^{-1}\times M_1$ and $\tilde{H}_{inv}{}^T = M_1\times\left(H^T\right)^{-1}$, therefore

$$\begin{aligned} e-\hat{e} &= \frac{1}{64}\left(H^{-1}\times 64\times E\times\left(H^T\right)^{-1} - H^{-1}\times M_1\times\hat{E}\times M_1\times\left(H^T\right)^{-1}\right) \\ &= \frac{1}{64}\left(H^{-1}\times\left(64\times E - M_1\times\hat{E}\times M_1\right)\times\left(H^T\right)^{-1}\right) \end{aligned} \tag{13}$$

Let

$$Y = 64\times E - M_1\times\hat{E}\times M_1, \tag{14}$$

Then substitute (13) and (14) into (10), we obtain

$$\begin{aligned} D &= trace\left((e-\hat{e})\times(e-\hat{e})^T\right) \\ &= trace\left(\frac{1}{64^2}\left(H^{-1}\times Y\times\left(H^T\right)^{-1}\times H^{-1}\times Y^T\times\left(H^T\right)^{-1}\right)\right) \end{aligned} \tag{15}$$

Let $M_2 = \left(H^T\right)^{-1}\times H^{-1} = diag(0.25,0.1,0.25,0.1)$, we also have $\left(H^T\right)^{-1} = M_2\times H$, so (15) becomes

$$D = trace\left(\frac{1}{64^2}\left(H^{-1}\times Y\times M_2\times Y^T\times M_2\times H\right)\right)$$
$$= \frac{1}{64^2}trace\left(Y\times M_2\times Y^T\times M_2\right) \quad (16)$$

Expand equation (16), we obtain

$$D = \frac{1}{64^2}\left(\begin{array}{l}\frac{1}{16}\times\left(Y(1,1)^2+Y(1,3)^2+Y(3,1)^2+Y(3,3)^2\right)+\\ \frac{1}{100}\times\left(Y(2,2)^2+Y(2,4)^2+Y(4,2)^2+Y(4,4)^2\right)+\\ \frac{1}{40}\times\left(\begin{array}{l}Y(1,2)^2+Y(1,4)^2+Y(2,1)^2+Y(4,1)^2\\ +Y(2,3)^2+Y(3,2)^2+Y(3,4)^2+Y(4,3)^2\end{array}\right)\end{array}\right) \quad (17)$$

Therefore, the distortion then can be determined from equation (17), where *Y* is give by equation (14).

Note that there may be a small difference between the above-described transform-domain distortion and the pixel-domain distortion due to the following factors. The inverse HT specified in the H.264/AVC specification is not strictly linear because an integer shift operation is used to realize the division-by-two. Another factor is the clipping in the reconstruction of the macroblock partition. However, the difference is expected to be negligible.

IV. ONGOING WORK

We have verified using MATLAB and the H.264/AVC reference software [5] that the distortion calculated in transform-domain is very accurate. We are currently implementing the proposed approach based on the reference software. We will provide more detailed performance evaluation once we have more results.

REFERENCES

[1] T. Wiegand et al., "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560-576, 2003.
[2] G. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Processing Magazine*, vol. 15, no. 6, pp. 74-90, Nov. 1998.
[3] T. Wiegand et al., "Rate-constrained coder control and comparison of video coding standards," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 688-703, July 2003.
[4] J. Xin, A. Vetro and H. Sun, "Converting DCT coefficients to H.264/AVC transform coefficients," (submitted to) *Pacific-rim Conference on Multimedia* (*PCM*), 2004.
[5] Available online: *http://bs.hhi.de/~suehring/tml/*.