Enhanced Transform Domain Intra Prediction for MPEG-2 to H.264/AVC Transcoding

Sang-Jun Yu and Chae-Bong Sohn

VIA-Multimedia Center, Kwangwoon University, 447-1, Wolgye-Dong, Nowon-Gu, 139-701, Seoul, Korea

Summary
In this paper, we proposed an enhanced transform domain intra prediction for H.264/AVC. In the proposed scheme, H.264 intra prediction can be described by generalized linear operation after defining both H-filter and V-filter. Since intra prediction operation can be decomposed into some sparse matrices, we can reduce computational complexity for intra prediction. Our scheme can reduce the number of multiplication and addition operations more than Chen's by 22.4% and 21.6%, respectively, in each 4×4 block. We can insist that the proposed scheme can be used for fast transform domain intra prediction in the area of transcoding MPEG-2 streams into H.264/AVC streams.

Key words:
H.264, MPEG-2, Intra Prediction, Transcoding

1. Introduction

In recent years, with the demand of multimedia for Internet, wireless communication and High Definition Television (HDTV), we need a powerful and high performance video compression standard, H.264/MPEG-4 Part 10 Advanced Video Coding (AVC). Especially, the research of transcoding that changes MPEG-2 video streams into H.264/AVC video streams has been being processed [1][2].

H.264/AVC is the newest video coding standard jointly developed by the ITU-T VCEG (Video Coding Experts Group) and the ISO/IEC MPEG (Moving Picture Experts Group). It provides better compression efficiency than previous video coding standards adopted some new coding technologies. However, it requires high computational complexity. Intra prediction is one of the key technologies in H.264/AVC. In contrast to some previous coding standards such as H.263+ and MPEG-4 Part-2, where intra prediction is performed in the transform domain, the intra prediction of H.264 is completely defined in the pixel domain by referring to neighboring samples of the coded blocks [3][4][5]. Thus, it is very difficult to convert an MPEG-2 stream into an H.264/AVC stream in the transform domain. To solve this problem, Chen proposed a transform domain intra prediction [3]. However, since his method is not fully optimized, we enhance Chen's method in terms of computational complexity in this paper.

2. H.264/AVC Intra Prediction and Previous Works

The intra prediction of 4×4 blocks defined by H.264/AVC is illustrated in Fig. 1(a), where the symbols a to p denote the pixels of current block, and the symbols A to M denote the neighboring pixels based on which the prediction block is calculated, The direction of prediction is shown in Fig. 1(b).

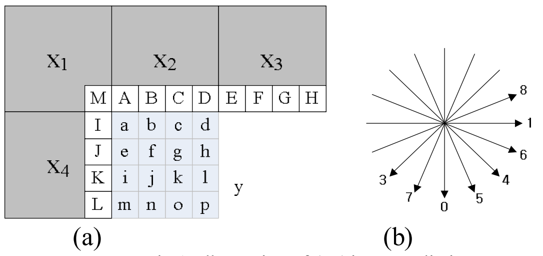

Fig.1. Illustration of 4×4 intra prediction

The intra prediction process can be described mathematically as a sequence of matrix multiplication [6].

Take mode 0 (vertical prediction) [2] as an example, we can represent the prediction of the current block as:

$$\begin{bmatrix} A & B & C & D \\ A & B & C & D \\ A & B & C & D \\ A & B & C & D \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ A & B & C & D \end{bmatrix} \qquad (1)$$

where the matrix on the left hand side of Eq. (1) represents the predicted block, the second matrix on the right hand side of Eq. (1) represents the upper block above the current block, and the symbol × means "don't care" pixel since we only need the bottom four pixels of the upper block in this prediction mode. The first matrix on the right hand side of Eq. (1) is called an operation

Manuscript received December 5, 2007
Manuscript revised December 20, 2007

matrix.

Then we apply the modified DCT transform to both sides of Eq. (1). Because modified DCT is a linear transformation, we can rewrite the right side of the above equation after the modified DCT transform as:

$$\begin{bmatrix} 1 & -1.265 & 1 & -0.632 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} X2_{11} & X2_{12} & X2_{13} & X2_{14} \\ X2_{21} & X2_{22} & X2_{23} & X2_{24} \\ X2_{31} & X2_{32} & X2_{33} & X2_{34} \\ X2_{41} & X2_{42} & X2_{43} & X2_{44} \end{bmatrix} \quad (2)$$

where $X2_{ij}$'s are the modified DCT coefficients of the upper block. For computational efficiency of the transcoding process, the operating matrix can be pre-computed and stored in memory. As we can see, 8 multiplications and 12 additions are required to perform this prediction mode in transform domain for a 4x4 block.

For mode 1 (horizontal) and mode 2 (DC) of the intra prediction [2], their transform-domain predictions can be derived in the same way. Except the first 3 modes, the transform-domain prediction processes of the remaining modes are relatively complicated. Due to the page limit, we only illustrate the transform-domain prediction for mode 3 (diagonal_down_left) here. All the remaining 4x4 intra prediction modes can be derived in a similar way.

The prediction performed in mode 3 can be described as follows:

$$\frac{1}{4}\begin{bmatrix} A+2B+C & B+2C+D & C+2D+E & D+2E+F \\ B+2C+D & C+2D+E & D+2E+F & E+2F+G \\ C+2D+E & D+2E+F & E+2F+G & F+2G+H \\ D+2E+F & E+2F+G & F+2G+H & G+3H \end{bmatrix} \quad (3)$$

Dividing the above equation into two terms according to the source of the pixel (from the upper block or from the upper-right block) yields:

$$= \begin{bmatrix} A+2B+C & B+2C+D & C+2D & D \\ B+2C+D & C+2D & D & 0 \\ C+2D & D & 0 & 0 \\ D & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & E & 2E+F \\ 0 & E & 2E+F & E+2F+G \\ E & 2E+F & E+2F+G & F+2G+H \\ 2E+F & E+2F+G & F+2G+H & G+3H \end{bmatrix} \quad (4)$$

We further divide each term into four rows. The first row of the first term can be rewritten as follows:

$$\begin{bmatrix} A+2B+C & B+2C+D & C+2D & D \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ A & B & C & D \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ 0 & 1 & 2 & 1 \end{bmatrix} \quad (5)$$

Where the symbol "$\times$" represents "don't care" pixel.

Similarly, other rows can be determined. The final value of the prediction is obtained by adding all these terms together.

The summation of the four blocks gives rise to the first matrix on the right-hand side of (4). The second matrix on the right-hand side of (4) can be calculated in a similar way.

At the first sight, it may seem that this intra prediction mode involves a lot of matrix multiplications. A closer look at the above four equations indicates that the first matrices on the right-hand side of (5) have a vertical shift relationship. We exploit such relationship to simplify the computation.

First, take the modified transform of the first matrices described above and denote them by $\mathbf{U}_1$, $\mathbf{U}_2$, $\mathbf{U}_3$, and $\mathbf{U}_4$, respectively:

$$\mathbf{U}_1 = \mathrm{HT}\left(\begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}\right), \mathbf{U}_2 = \mathrm{HT}\left(\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}\right), \mathbf{U}_3 = \mathrm{HT}\left(\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}\right), \mathbf{U}_4 = \mathrm{HT}\left(\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\right) \quad (6)$$

Denoting the modified DCT used in H.264 by HT( ). Then, multiply U1 with the modified DCT of the upper block and denote the resulting matrix by Y1.

We thus have:

$$\mathbf{Y}_1 = \mathbf{U}_1\mathbf{X} = \begin{bmatrix} a & -b & a & c \\ 1.264a & -1.264b & 1.264a & 1.264c \\ a & -b & a & c \\ 0.632a & -0.632b & 0.632a & 0.632c \end{bmatrix} \qquad (7)$$

where a = 0.25, b=0.316, c=0.159, and

$$\mathbf{X} = \mathrm{HT}\left(\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ A & B & C & D \end{bmatrix}\right) = \begin{bmatrix} X_{11} & X_{12} & X_{13} & X_{14} \\ X_{12} & X_{22} & X_{23} & X_{24} \\ X_{13} & X_{32} & X_{33} & X_{34} \\ X_{14} & X_{42} & X_{43} & X_{44} \end{bmatrix} \qquad (8)$$

Let

$$\begin{aligned} S_0 &= a(X_{11} + X_{31}) - (bX_{21} + cX_{41}) \\ S_1 &= a(X_{12} + X_{32}) - (bX_{22} + cX_{42}) \\ S_2 &= a(X_{13} + X_{33}) - (bX_{23} + cX_{43}) \\ S_3 &= a(X_{14} + X_{34}) - (bX_{24} + cX_{44}) \end{aligned} \qquad (9)$$

Then, (7) can be rewritten as:

$$\mathbf{Y}_1 = \begin{bmatrix} S_0 & S_1 & S_2 & S_3 \\ 1.264S_0 & 1.264S_1 & 1.264S_2 & 1.264S_3 \\ S_0 & S_1 & S_2 & S_3 \\ 0.632S_0 & 0.632S_1 & 0.632S_2 & 0.632S_3 \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{Y}}_1 \\ \tilde{\mathbf{Y}}_2 \\ \tilde{\mathbf{Y}}_1 \\ \tilde{\mathbf{Y}}_3 \end{bmatrix} \qquad (10)$$

where, $\tilde{\mathbf{Y}}_i, 1 \le i \le 3$, denote different row vectors of $\mathbf{Y}_1$. Note that the computation of the matrix $\mathbf{Y}_1$ is now reduced to 20 multiplications and 12 additions. Because of the vertical shift relationship, the remaining three matrices $\mathbf{Y}_2 = \mathbf{U}_2\mathbf{X}$, $\mathbf{Y}_3 = \mathbf{U}_3\mathbf{X}$, and $\mathbf{Y}_4 = \mathbf{U}_4\mathbf{X}$ are also composed of $\tilde{\mathbf{Y}}_1$, $\tilde{\mathbf{Y}}_2$, and $\tilde{\mathbf{Y}}_3$

$$\mathbf{Y}_2 = \mathbf{U}_2\mathbf{X} = \begin{bmatrix} \tilde{\mathbf{Y}}_1 \\ \tilde{\mathbf{Y}}_3 \\ -\tilde{\mathbf{Y}}_1 \\ -\tilde{\mathbf{Y}}_2 \end{bmatrix}, \mathbf{Y}_3 = \begin{bmatrix} \tilde{\mathbf{Y}}_1 \\ -\tilde{\mathbf{Y}}_3 \\ -\tilde{\mathbf{Y}}_1 \\ \tilde{\mathbf{Y}}_2 \end{bmatrix}, \mathbf{Y}_4 = \begin{bmatrix} \tilde{\mathbf{Y}}_1 \\ -\tilde{\mathbf{Y}}_2 \\ \tilde{\mathbf{Y}}_1 \\ -\tilde{\mathbf{Y}}_3 \end{bmatrix}, \qquad (11)$$

Therefore, $\mathbf{Y}_2$, Y3, and Y4 are readily available without any additional calculation. Moreover, since the matrices Y1, Y2, Y3, and Y4 are needed in the prediction process of other modes, we can store them in memory to avoid unnecessary computation. At the first look, it may seem that the matrix multiplications involved in the transform domain prediction would involve lots of computations, but in fact it can be further simplified because the operating matrices of each row have a vertical shift relationship [6].

3. Enhanced Transform Domain Intra Prediction (ETDIP)

In this section, we propose an enhanced transform domain intra prediction. In case of mode 0(vertical prediction) as an example, we can represent the prediction of the current blocks as (1). The symbol "×" means a "don't care" pixel, which does not have an effect on the results. To reduce the computational complexity, we used only the bottom four pixels instead of "don't care" pixels. Therefore, (1) can be rewritten as (12):

$$\begin{bmatrix} A & B & C & D \\ A & B & C & D \\ A & B & C & D \\ A & B & C & D \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} A & B & C & D \\ A & B & C & D \\ A & B & C & D \\ A & B & C & D \end{bmatrix} \qquad (12)$$

Denoting the modified DCT used in H.264/AVC by HT and applying it both side of (12), we can rewrite the right side of (12) as (13) since HT is a linear transform [2]:

$$\begin{bmatrix} 1 & -1.265 & 1 & -0.632 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_1 & V_2 & V_3 & V_4 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad (13)$$

where V_i's are the modified DCT coefficients at the vertical prediction block. Compare with (2) and (13), as we can see, it has only four nonzero coefficients. Using this property we can easily simplify other intra prediction modes.

In the proposed scheme, H.264 intra prediction can be described by generalized linear operation after defining both H-filter and V-filter since intra prediction operation can be decomposed into some sparse matrices.
First, we defined H-filter and V-filter as (14).

$$\mathrm{H}(\mathbf{X}) = \mathbf{X}\left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}\right), \quad \mathrm{V}(\mathbf{X}) = \left(\begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}\right)\mathbf{X} \qquad (14)$$

Then, intra prediction process can be generalized as (15).

$$\mathbf{y}^m_{\mathrm{pred}} = \left(\sum_{i=1}^{4} \mathbf{s}_i \mathbf{x}_1 \mathbf{c}^m_{1,i}\right) + \left(\sum_{n=2}^{3}\sum_{i=1}^{4} \mathbf{s}_i \mathbf{x}_n \mathbf{c}^m_{n,i}\right) + \left(\sum_{i=1}^{4} \mathbf{c}^m_{4,i} \mathbf{x}_4 \mathbf{s}_i^{\mathrm{T}}\right) \qquad (15)$$

where $\mathbf{x}_n$ is the n-th neighboring block that is used to predict a current block y, $\mathbf{c}^m_{n,i}$ is a constant table according to intra prediction mode, and $\mathbf{s}_i$ is a shifting matrix as following:

$$\mathbf{s}_i = (a_{ij})_{4\times 4}, \quad a_{ij} = \begin{cases} 1, if\ j = 4\ and\ i = \alpha\ (1 \le \alpha \le 4) \\ 0, else \end{cases}$$

According to (14), Equation (15) can be rewritten as (16).

$$\mathbf{y}^m_{\text{pred}} = \left(\sum_{i=1}^{4}\mathbf{s}_i \mathrm{V}(\mathrm{H}(\mathbf{x}_1))\mathbf{c}^m_{1,i}\right) + \left(\sum_{i=1}^{4}\mathbf{s}_i \mathrm{V}(\mathbf{x}_2)\mathbf{c}^m_{2,i}\right) + \left(\sum_{i=1}^{4}\mathbf{s}_i \mathrm{V}(\mathbf{x}_3)\mathbf{c}^m_{3,i}\right) + \left(\sum_{i=1}^{4}\mathbf{c}^m_{4,i}\mathrm{H}(\mathbf{x}_4)\mathbf{s}_i^{\mathrm{T}}\right) \quad (16)$$

Equation's (15) and (16) show that intra prediction can be decomposed into four blocks with neighboring pixels. Each block can be calculated using pre-computed either H or V-filter coefficients.

For computational efficiency of the proposed method, we analyze the transform-domain intra prediction in general case.

In case of the n-th neighboring 4×4 block and mode m, the transform-domain intra predicted 4×4 block can be expressed as (17) applying HT into both sides of (16).

$$\mathrm{HT}\left(\sum_{i=1}^{4}\mathbf{s}_i \mathrm{V}(\mathbf{x}_n)\mathbf{c}^m_{n,i}\right) = \mathrm{HT}\left(\mathbf{s}_1 \mathrm{V}(\mathbf{x}_n)\mathbf{c}^m_{n,1}\right) + \mathrm{HT}\left(\mathbf{s}_2 \mathrm{V}(\mathbf{x}_n)\mathbf{c}^m_{n,2}\right) + \mathrm{HT}\left(\mathbf{s}_3 \mathrm{V}(\mathbf{x}_n)\mathbf{c}^m_{n,3}\right) + \mathrm{HT}\left(\mathbf{s}_4 \mathrm{V}(\mathbf{x}_n)\mathbf{c}^m_{n,4}\right) \quad (17)$$

Then each term can be calculated without constant tables as follows:

$$\begin{aligned}
\mathrm{HT}(\mathbf{s}_1\ \mathrm{V}(\mathbf{x}_n)) &= \mathrm{HT}\left(\begin{bmatrix}0&0&0&1\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{bmatrix}\right)\mathrm{HT}\left(\begin{bmatrix}A&B&C&D\\A&B&C&D\\A&B&C&D\\A&B&C&D\end{bmatrix}\right)\\
&= \begin{bmatrix}a&-b&a&-c\\1.264a&-1.264b&1.264a&-1.264c\\a&-b&a&-c\\0.632a&-0.632b&0.632a&-0.632c\end{bmatrix}\begin{bmatrix}V_1&V_2&V_3&V_4\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{bmatrix}\\
&= a\times\begin{bmatrix}V_1&V_2&V_3&V_4\\1.264V_1&1.264V_2&1.264V_3&1.264V_4\\V_1&V_2&V_3&V_4\\0.632V_1&0.632V_2&0.632V_3&0.632V_4\end{bmatrix}\\
&= a\times\begin{bmatrix}\tilde{\mathbf{Y}}_{n,1}\\1.264\tilde{\mathbf{Y}}_{n,1}\\\tilde{\mathbf{Y}}_{n,1}\\0.632\tilde{\mathbf{Y}}_{n,1}\end{bmatrix}
\end{aligned} \quad (18)$$

where $\tilde{\mathbf{Y}}_{n,1} = [V_1\ V_2\ V_3\ V_4]_{1\times4}$. $\tilde{\mathbf{Y}}_{n,1}$ is pre-computed V-filter coefficients. Chen's method needs additional operations to matrix calculations which include "don't care" value to get matrix Y. But, our method can avoid those calculations using H/V filter. Especially, we can significantly reduce computational complexity through horizontal and vertical filtering using pre-computed coefficients for each mode.

Similarly, other terms in (17) can be determined. Because of the vertical shift relationship, the remaining three terms are also composed of $\tilde{\mathbf{Y}}_{n,1}$ [3]. The final value of the prediction is obtained by adding all these terms together.

Finally, we obtained transform-domain intra prediction matrix of (17) as following:

$$\mathbf{Y}^m_{\text{pred},n} = \mathrm{HT}\left(\mathbf{y}^m_{\text{pred},n}\right) = \begin{bmatrix}\left(\tilde{\mathbf{Y}}_{n,1}\right)_{1\times4}\times\left(\tilde{\mathbf{C}}^m_{n,1}\right)_{4\times4}\\\left(\tilde{\mathbf{Y}}_{n,1}\right)_{1\times4}\times\left(\tilde{\mathbf{C}}^m_{n,2}\right)_{4\times4}\\\left(\tilde{\mathbf{Y}}_{n,1}\right)_{1\times4}\times\left(\tilde{\mathbf{C}}^m_{n,3}\right)_{4\times4}\\\left(\tilde{\mathbf{Y}}_{n,1}\right)_{1\times4}\times\left(\tilde{\mathbf{C}}^m_{n,4}\right)_{4\times4}\end{bmatrix} \quad (19)$$

where $\tilde{\mathbf{C}}^m_{n,i}, 1\le i\le 4$,

$$\begin{aligned}
\tilde{\mathbf{C}}^m_{n,1} &= a\times\left(\mathbf{C}^m_{n,1}+\mathbf{C}^m_{n,2}+\mathbf{C}^m_{n,3}+\mathbf{C}^m_{n,4}\right)\\
\tilde{\mathbf{C}}^m_{n,2} &= a\times\left(b\times\left(\mathbf{C}^m_{n,1}-\mathbf{C}^m_{n,4}\right)+c\times\left(\mathbf{C}^m_{n,2}-\mathbf{C}^m_{n,3}\right)\right)\\
\tilde{\mathbf{C}}^m_{n,3} &= a\times\left(\mathbf{C}^m_{n,1}-\mathbf{C}^m_{n,2}-\mathbf{C}^m_{n,3}+\mathbf{C}^m_{n,4}\right)\\
\tilde{\mathbf{C}}^m_{n,4} &= a\times\left(c\times\left(\mathbf{C}^m_{n,1}-\mathbf{C}^m_{n,4}\right)+b\times\left(\mathbf{C}^m_{n,3}-\mathbf{C}^m_{n,2}\right)\right)
\end{aligned}$$

Thus, only 64 multiplications and 48 additions are required for each 4×4 block in the transform-domain.

Similarly, other terms in (16) can be determined. Especially, in case of the first term of (16), we can achieve more reduced computational complexity than the others using a property of H-filter and V-filter in same time.
The first term of (16) can be rewritten as:

$$\mathbf{y}^m_{\text{pred},1} = \sum_{\mathrm{i}=1}^{4}\mathbf{s}_{\mathrm{i}}\mathrm{H}\left(\mathbf{x}_{1,v}\right)\mathbf{c}^m_{1,\mathrm{i}} = \mathbf{s}_1\mathrm{H}\left(\mathbf{x}_{1,v}\right)\mathbf{c}^m_{1,1} + \mathbf{s}_2\mathrm{H}\left(\mathbf{x}_{1,v}\right)\mathbf{c}^m_{1,2} + \mathbf{s}_3\mathrm{H}\left(\mathbf{x}_{1,v}\right)\mathbf{c}^m_{1,3} + \mathbf{s}_4\mathrm{H}\left(\mathbf{x}_{1,v}\right)\mathbf{c}^m_{1,4} \quad (20)$$

The transform-domain intra predicted 4×4 block can be expressed as (21) applying HT into both sides of (20) without constant table.

$$
\begin{aligned}
\mathrm{HT}\left(\mathbf{s}_1\,\mathrm{H}\left(\mathbf{x}_{1,v}\right)\right) &= \mathrm{HT}\left(\begin{bmatrix}0&0&0&1\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{bmatrix}\mathrm{H}\left(\begin{bmatrix}A&B&C&D\\A&B&C&D\\A&B&C&D\\A&B&C&D\end{bmatrix}\right)\right)\\
&= \mathrm{HT}\left(\begin{bmatrix}0&0&0&1\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{bmatrix}\right)\left(\begin{bmatrix}A&B&C&D\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{bmatrix}\begin{bmatrix}1&0&0&0\\-1.264&0&0&0\\1&0&0&0\\-0.632&0&0&0\end{bmatrix}\right)\\
&= \begin{bmatrix}a&-b&a&-c\\1.264a&-1.264b&1.264a&-1.264c\\a&-b&a&-c\\0.632a&-0.632b&0.632a&-0.632c\end{bmatrix}\begin{bmatrix}Q&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{bmatrix} \qquad (21)\\
&= a\times\left(Q\times\begin{bmatrix}1&0&0&0\\1.264&0&0&0\\1&0&0&0\\0.632&0&0&0\end{bmatrix}\right) = a\times Q\times\begin{bmatrix}\widetilde{\mathbf{Y}}_{1,1}\\1.264\widetilde{\mathbf{Y}}_{1,1}\\\widetilde{\mathbf{Y}}_{1,1}\\0.632\widetilde{\mathbf{Y}}_{1,1}\end{bmatrix}
\end{aligned}
$$

where $Q = (A + C) - 0.632(2B + D)$, and $\widetilde{\mathbf{Y}}_{1,1} = [1\ 0\ 0\ 0]_{1\times 4}$.

In this case we only required 3 additions and 2 multiplications.

Using a similar procedure described previously, other terms in (20) can be determined. Finally, we obtained transform-domain intra prediction matrix of (20) as following:

$$
\mathbf{Y}^{m}_{\mathrm{pred},1} = \mathrm{HT}\left(\mathbf{y}^{m}_{\mathrm{pred},1}\right) = \begin{bmatrix}\left(\widetilde{\mathbf{Y}}_{1,1}\right)_{1\times4}\times\left(\widetilde{\mathbf{C}}^{m}_{1,1}\right)_{4\times4}\\\left(\widetilde{\mathbf{Y}}_{1,1}\right)_{1\times4}\times\left(\widetilde{\mathbf{C}}^{m}_{1,2}\right)_{4\times4}\\\left(\widetilde{\mathbf{Y}}_{1,1}\right)_{1\times4}\times\left(\widetilde{\mathbf{C}}^{m}_{1,3}\right)_{4\times4}\\\left(\widetilde{\mathbf{Y}}_{1,1}\right)_{1\times4}\times\left(\widetilde{\mathbf{C}}^{m}_{1,4}\right)_{4\times4}\end{bmatrix} \qquad (22)
$$

where $\widetilde{\mathbf{C}}^{m}_{1,i}, 1 \le i \le 4$,

$$
\begin{aligned}
\widetilde{\mathbf{C}}^{m}_{1,1} &= a\times\left(\mathbf{C}^{m}_{1,1}+\mathbf{C}^{m}_{1,2}+\mathbf{C}^{m}_{1,3}+\mathbf{C}^{m}_{1,4}\right)\\
\widetilde{\mathbf{C}}^{m}_{1,2} &= a\times\left(b\times\left(\mathbf{C}^{m}_{1,1}-\mathbf{C}^{m}_{1,4}\right)+c\times\left(\mathbf{C}^{m}_{1,2}-\mathbf{C}^{m}_{1,3}\right)\right)\\
\widetilde{\mathbf{C}}^{m}_{1,3} &= a\times\left(\mathbf{C}^{m}_{1,1}-\mathbf{C}^{m}_{1,2}-\mathbf{C}^{m}_{1,3}+\mathbf{C}^{m}_{1,4}\right)\\
\widetilde{\mathbf{C}}^{m}_{1,4} &= a\times\left(c\times\left(\mathbf{C}^{m}_{1,1}-\mathbf{C}^{m}_{1,4}\right)+b\times\left(\mathbf{C}^{m}_{1,3}-\mathbf{C}^{m}_{1,2}\right)\right)
\end{aligned}
$$

4. Computational Complexity

In this section, we analyze the computational complexity of both the proposed and Chen's methods in terms of the number of operations required to obtain the prediction same.

The number of operations required for each intra prediction mode is shown in Table I. The table shows that our method has lower computational complexity than Chen's from modes 3 to 6. Our scheme can reduce the number of multiplication and addition operations more than Chen's by 22.4% and 21.6%, respectively, in each 4×4 block.

Table 1: Computational Complexity of 4×4 Intra Prediction

Method / Mode	Chen's method[3]		ETDIP	
	Mult.	Add.	Mult.	Add.
0	8	12	8	12
1	8	12	8	12
2	1	1	1	1
3	168	136	128	112
4	232	200	146	131
5	192	176	144	128
6	192	176	144	128
7	128	112	128	112
8	64	48	64	48
Total	993	873	771	684
Imp. (%)			22.4%	21.6%

5. Conclusion

In this paper, we proposed an enhanced transform domain intra prediction for H.264/AVC. Since intra prediction operation can be decomposed into some sparse matrices, we can reduce computational complexity. Our scheme can reduce the number of operations more than Chen's by 22% in each 4×4 block.

Consequently, we can insist that the proposed scheme can be used for fast transform domain intra prediction in the area of transcoding MPEG-2 streams into H.264/AVC streams.

Acknowledgments

The present Research has been conducted by the Research Grant of Kwangwoon University in 2007.

References

[1] ISO/IEC 13818-2, "Information technology-Generic coding of moving pictures and associated audio information: Video," 1995.

[2] ITU-T Rec. H.264|ISO/IEC 14496-10, "Advanced Video Coding," 2003.
[3] C. Chen, Ping-Hao Wu and H. Chen, "Transform-Domain Intra Prediction for H.264," *IEEE ISCAS*, pp. 1497-1500, May 2005.
[4] A. Vetro, C. Christopoulos, and H. Sun, "Video transcoding architectures and techniques: An overview," IEEE Signal Processing Magazine, pp 18-29, Mar. 2003.
[5] T. Qian, J. Sun, D. Li, Yang X. and J. Wang, "Transform domain transcoding from MPEG-2 to H.264 with interpolation drift-error compensation," *CSVT, IEEE Transaction* on vol.16, Issue 4, pp.523-534, Apr. 2006.
[6] Chen Chen, Ping-Hao Wu and Homer Chen, "MPEG-2 to H.264 Transcoding", *Proc. Picture Coding Symposium,* Dec. 2004.

Sang-Jun Yu received the B.S. and M.S., degree in electronic engineering from Kwangwoon University, Seoul, Korea in 1998, 2003, respectively. He is currently a Ph.D. student in the Multimedia Laboratory at the University of Kwangwoon, Seoul, Korea. He was a part-time lecturer in University of Kwangwoon at 2006 and 2007, Department of Electronic Engineering. His research interests include image compression, video coding and transcoding.

Chae-Bong Sohn received the B.S., M.S., and Ph.D. degree in electronics engineering from Kwangwoon University, Seoul, Korea in 1993, 1995, and 2006, respectively. From 2000 to 2005, he worked in Hanyang Women's College as a full-time lecturer. He is currently an assistant professor in department of Electronics and Communications Engineering, Kwangwoon University. His research interests include image compression, transcoding, digital broadcasting systems.