COMPLEXITY REDUCTION OF IDCT BY USING CBP AND ADAPTIVE PRUNING IN H.264/AVC DECODER

Luong Pham Van\textsuperscript{1} and Byeungwoo Jeon\textsuperscript{2}
School of Electronic and Electrical Engineering
Sungkyunkwan University
300 Chunchundong, Janganggu, Suwon, Korea
\textsuperscript{1}phamvanluongdt12k49@gmail.com and \textsuperscript{2}bjeon@skku.edu

ABSTRACT
In this paper, we propose two methods to reduce computational complexity of H.264/AVC decoder. The proposed methods utilize CBP (Coded Block Pattern) information to skip inverse DCT transform and reconstruction processes for 8x8 blocks that have CBP bits being zero. Furthermore, we adaptively skip zero coefficients in a block to reduce unnecessary computation in inverse DCT transform module. Experimental results show that the proposed method can reduce decoding complexity by about 10 \% with only a negligible PSNR drop.

KEY WORDS
Decoder complexity, H.264/AVC, inverse transform, CBP.

1. Introduction
The H.264/AVC (Advanced Video Coding) coding standard achieved excellent compression performance, and its high compression performance made it attractive for wide range of applications from mobile communication to HDTV. However, to achieve its high compression performance, it adopted many advanced coding tools such as intra prediction, motion compensation up to quarter pel, adaptive deblocking filter, and integer discrete cosine transform, etc. Due to these coding tools, its computational complexity is significantly increased at both encoder and decoder. Its high computational complexity can be a practical problem for many low-cost mobile and real time applications which in generally have low processing power. For this reason, the development of real time and mobile application makes the optimizing H.264/AVC a very popular research topic.

Many methods have been already researched in order to reduce computational complexity of H.264/AVC decoder. We can classify these approaches into two categories. The first one is to solve the problem by optimizing the performance of hardware. K. Shin et al. [1] built a state-of-the-art parallel H.264 decoder using symmetric multicore processors like Intel Core2Quad and ARM MPCore processor. This method exploited the parallel processing capability of multicore processors to speed up the decoding time. In addition to this category, S. Yang et al. [2] segmented a frame into N rectangles then each of which was given in to one-core among N-cores for deblocking filter. Each segment can be deblocked concurrently. Those methods in the second category focus on exploiting characteristic of video sequence to reuse some information in order to reduce complexity of decoder. In [3], H. Chen optimized the BS (Boundary Strength) value decision for deblocking by exploiting the correlation among BS values of internal edges in a large macroblock. Most existing methods have focused on optimizing deblocking filter and motion compensation since they are the most complex part of a decoder. Therefore their optimization can significantly reduce the processing complexity. If further reduction of decoder complexity is sought, one candidate module to smartly manipulate is the inverse DCT transform (IDCT) module. S. Malvar et al. [4] applied integer transform for 4x4 DCT (Discrete Cosine Transform). The transform can be computed with just additions and shifts in 16-bit arithmetic - that is, without any multiplication, thus minimizing computation. Note that the DCT coefficients at high frequencies are very likely to be zero, and it still takes complexity to manipulate such zero-coefficients. In [5], S. Peng et al. used a pruning method for DCT data - they set some rows and/or columns zero at high frequencies to omit unnecessary computation. However, such zero setting of non-zero coefficients deteriorates quality of decoded frames.

In this paper, we propose two simplification methods of IDCT for decoder complexity reduction. First, we use the CBP (coded block pattern) information to early decide whether or not to skip IDCT and reconstruction modules altogether. Secondly we adaptively prune DCT coefficients by checking the number of last zero coefficients in 4x4 DCT coefficients. Experimental results show that the proposed methods can reduce computational complexity of decoder by about 10 \%.

The rest of the paper is as follows. Section 2 describes the meaning of CBP and how to use it to computationally optimize a H.264/AVC decoder. Section 3 explains the proposed adaptive pruning method to reduce complexity of IDCT module and the combination of the two proposed methods. Experimental results are given in Section 4 and conclusion is given in Section 5.
2. Decoder Optimization Using CBP

The coded block pattern (CBP) in H.264/AVC is a syntax element in the macroblock layer, which specifies whether each of the six 8x8 blocks contains non-zero transform coefficients or not [6]. The CBP consists of six bits “C5C4C3C2C1C0” (Fig. 1). The last four bits (C3C2C1C0) individually give the coded block pattern information of the luma blocks (Y), and the other two bits (C5C4) jointly give information about the chroma blocks (Cb and Cr). Here, a zero bit indicates that all DCT coefficients of corresponding four 4x4 blocks in the given 8x8 block are zero, thus the four 4x4 blocks are not coded. For example, if the CBP is 010010, one 8x8 Y block and one Cb block have been coded. For the reconstruction of the non-coded blocks, the IDCT and reconstruction modules are not invoked, instead, the decoder only needs to copy predicted values to corresponding reconstructed blocks.

The JM17.0 reference software [7] has an implementation in which the IDCT module is called as following. At first, it checks the four bits of CBP corresponding to the luma blocks. If at least one bit is non-zero, the IDCT and reconstruction modules are called for all luma blocks (that is, 16 4x4 blocks) of the macroblock although some of them may have all zero DCT coefficients especially in P- or B-slices or in low bitrate case. Among those four bits of CBP, in many cases, they may have only 1 or 2 non-zero bits and the other bits are zero. As mentioned above, when a bit in the CBP bits is zero, the IDCT and reconstruction modules need not be activated for the block corresponding to the zero CBP bit. Note that 4x4 IDCT operation (will be discussed below) includes 64 additions and 16 shifts [8]. Reconstructing one 8x8 block requires calling the 4x4 IDCT function four times, which means 256 additions and 64 shifts. The reconstruction process of one pixel requires 2 comparisons, 2 shifts, and 3 additions. Therefore, reconstruction of one 8x8 block takes 128 comparisons, 128 shifts, and 192 additions. The total complexity of IDCT and reconstruction for one 8x8 block is 448 additions, 192 shifts, and 128 comparisons. Therefore, it is very wasteful if the two modules are called for a 8x8 block which has corresponding CBP bit being zero. To solve this problem, we optimize a decoder as following: we check each 6 bit of CBP; if a bit is non-zero, we call IDCT and reconstruction functions for four 4x4 blocks inside the 8x8 block, else we just copy the predicted values for the 8x8 block to the reconstructed frame. In the proposed decoder optimization method using CBP information, the complexity redundancy coming from the IDCT and the reconstruction of non-coded blocks are removed. In this paper, it is called OD-CBP (computationally Optimized Decoder by referring to CBP).

The decoder complexity is significantly reduced by this method without degrading PSNR. Although at least one bit of luma CBP is non-zero, however, some of its 4x4 luma blocks inside the 8x8 block corresponding to the non-zero CBP bit may still have all zero coefficients. In this case, we can further reduce the complexity redundancy of the IDCT module. This idea is extended in this paper by adaptive IDCT pruning. In the next section, we utilize the number of last zero coefficients to further reduce the complexity of H.264/AVC decoder.

3. Adaptive Pruning in H.264/AVC Decoder

In order to optimize a H.264/AVC decoder in complexity by adaptive pruning, we investigate characteristic of DCT coefficients first and then discuss a method to reduce IDCT complexity. To investigate the characteristics of the DCT coefficients, we did some mini-experiments with many QP values and video sequences to evaluate the probability of DCT coefficients being non-zero at each frequency location. Fig. 2 depicts one result of our observations which lead to a conclusion that the probability of non-zero DCT coefficients at high frequency is very small. In the DCT domain, typically DCT coefficients at high frequencies are not as significant as the ones at low frequencies. Therefore, the quantization step size for those coefficients is in general set high. Consequently, the probability that the DCT coefficients at high frequencies being zero is high. Based on this quite expected observation, our proposed adaptive pruning method searches for zero DCT coefficients at high frequencies first and then the coefficients at low frequencies. In H.264/AVC encoding, a 4x4 integer transform is applied to convert 4x4 residual blocks to DCT domain. As a result, an inverse transform is used to convert those DCT coefficients back to pixel domain to obtain residual values. The inverse transform can be computed as one 1-D row transforms followed by 1-D column transform. Each of 1-D transform is computed by the equations below [8]:

\[
\begin{align*}
M0 &= Y0 + Y2; & M1 &= Y0 - Y2; \\
M2 &= (Y1 >> 1) - Y3; & M3 &= Y1 + (Y3 >> 1); \\
X0 &= M0 + M3; & X3 &= M0 - M3; \\
X1 &= M1 + M2; & X2 &= M1 - M2;
\end{align*}
\]

From Fig. 3, we can see that, in order to compute 1-D row or column transform, the processor uses 8 additions and 2 shifts. To complete the inverse 4x4 block transform, the processor has to carry out 8 1-D row and column transforms.
transforms. The total complexity of the 4x4 inverse transform will be 64 additions and 16 shifts. When a coefficient is zero, we can skip some operations that are related to the zero coefficient. This is the key idea of the pruning DCT data. To skip unnecessary operations, we have to identify whether a coefficient at a particular location is zero or not.

The complexity of such check can be much larger than calculating residual with DCT coefficients. In [5], S. Peng did not check the condition and proposed some fixed pruning pattern that dedicated some rows and/or columns for zero. The complexity reduction of IDCT depends on the number of rows and columns assigned zero. For example, the last row and column are assigned by zero. The 1-D transform is modified as following:

\begin{align*}
M0 &= Y0 + Y2; & M1 &= Y0 - Y2; \\
M2 &= (Y1 >> 1); & M3 &= Y1; \\
X0 &= M0 + M3; & X3 &= M0 - M3; \\
X1 &= M1 + M2; & X2 &= M1 - M2;
\end{align*}

In this case, 4x4 IDCT needs only three 1-D transforms and four 1-D transforms, respectively for 3 rows and 4 columns - the complexity of IDCT is now 42 addition and 7 shifts. If both one addition and one shift are completed in one clock cycle, the IDCT will take 49 cycles. Compared to the conventional methods, the proposed pattern can save \((80-49)/80*100 = 38.75\%\) of IDCT complexity. By using this pruning method, inverse transform complexity can be significantly reduced. In [5], many non-zero coefficients are assigned zero. However, it severely degrades PSNR performance. To solve this problem, in this paper, we propose an adaptive pruning method. A 4x4 block has tendency to have many zero coefficients at high frequency locations. After the entropy (CABAC or CAVLC) decoding, a decoder should have already known the number of zero coefficients in the last parts of a block by \textit{EOB} (End Of Block) information. In order to adaptively choose the best pruning pattern for the given 4x4 block, we define \textit{NLZC} (number of the last zero coefficients). The best pruning pattern to choose for the given 4x4 block is the one that has the minimum computational complexity at the same time with zeroing only the smallest number of non-zero coefficients of DCT block in the pruning process. In one 4x4 DCT block, there can be total 15 values of \textit{NLZC}, however, the proposed method group them into 4 cases. There is an exception for the 15 cases - for the case of only DC is non-zero (that is, \textit{NLZC} = 15), a weighted DC value is copied as residual pixel values. Fig. 4 has a summary of the proposed best pattern corresponding to each \textit{NLZC}. The first reason for this grouping is to minimize memory space (and its associated complexity) increment due to storing 15 different functions of IDCT corresponding to the 15 cases. The second reason is to have additional flexibility in controlling the amount of IDCT complexity reduction by extra pruning. In this way, it is possible to prune out coefficients higher in frequency without much side-effect coming from nullifying DCT coefficients. In the proposed method we like to choose the best pruning pattern for each 4x4 block. If we check the number \textit{NLZC} of each block and then call the corresponding IDCT for the block, the numbers of "if" command should be increased.

![Figure 2. The probability of non-zero coefficients in DCT block based on its frequencies](image)

![Figure 3. 1-D Inverse Transform](image)
The complexity of "if" command is much larger than that of any other commands such as addition, shift, and etc. In addition, the "if" command can interfere with the pipelining in parallel processing. To solve the problem, we use a look-up-table and a function pointer. We generate a table of 16 function pointers which point to 4 IDCT functions of the four pruning patterns. When a call to IDCT module is made for a block, instead of checking the NLZC of the block, we rather use the NLZC as an index in the pointer table to call the corresponding IDCT function for the block. We have presented two optimization methods for reducing complexity of H.264/AVC decoding. The first one (**OD-CBP**) utilizes the CBP to remove complexity redundancy of the IDCT and the reconstruction blocks which have CBP being zero. The second one (**APR (Adaptive PRuning)**) adaptively prunes DCT coefficients out to reduce complexity of IDCT module. The combination of the two methods (**OD-APR**) is designed as following. Since the **OD-CBP** method does not have any impact on decoded picture quality at all, it is applied for all I-, P-, and B-slices. After that, the **APR** is applied for those 4x4 blocks that correspond to non-zero CBP bit. The number of I-slices is not large and their overall computational complexity is also very small. However, their decoded quality is very important because I-slices are referenced by P- and B-slices. In addition, the P-slices are referenced by the other P-slices and B-slices. Therefore, they are also very important. So in the I-slices and P-slices, we use only the **OD-CBP** method. For the B-slices, the proposed two methods (**OD-CBP** and **APR**) are both used in decoding. The **APR** method can lower the decoded picture quality slightly. However, the grouping of pruning patterns in response to **NLZC** minimizes its decoding quality loss.

### 4. Experimental Results

The proposed methods are implemented on H.264/AVC reference software JM 17.0 (main profile) decoder [7]. Four SD sequences (BigShips, Night, City, and Crew) with the resolution of 720x576 are used with 300 frames for each sequence. The quantization parameter **QP** is set to three values of 22, 27, and 32. The coding structure is IBBPBBP and GOP size is 60. Experiments are done under Window-XP OS, Intel(R) Core2Duo CPU2GHz.

The average saving time (**AST**) is computed as following:

$$ AST[\%] = \frac{T_{Proposed} - T_{JM}}{T_{JM}} \times 100 $$

$$ DPSNR[db] = (PSNRY_{Proposed} - PSNRY_{JM}) $$

where $T_{JM}$ and $T_{Proposed}$ are running time of JM17.0 and the proposed method, respectively. Experimental results of using only the adaptive pruning method are given in Table 1 which shows that the proposed method can significantly reduce the IDCT complexity - ranging from 49% to 62%. When **QP** is high, the number of zero coefficients is expected to be high (that is, high **NLZC**). Therefore, the number of non-zero coefficient which is set zero is small. Consequently, when **QP** is high, the PSNR drop is small as in Table 1. In addition, when the number of zero coefficients is high, the third and fourth pruning pattern in Fig. 4 with high of complexity reduction are used more so that the complexity of IDCT is reduced much. Table 1 backs up the conclusion that the higher **QP** is, the higher IDCT complexity reduction is, and the smaller the PSNR drop is.

<table>
<thead>
<tr>
<th><strong>QP</strong></th>
<th>Bigship</th>
<th><strong>△PSNR[db]</strong></th>
<th><strong>AST[%]</strong></th>
<th><strong>△PSNR[db]</strong></th>
<th><strong>AST[%]</strong></th>
<th><strong>△PSNR[db]</strong></th>
<th><strong>AST[%]</strong></th>
<th><strong>△PSNR[db]</strong></th>
<th><strong>AST[%]</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>-0.38</td>
<td>56.07</td>
<td>-0.40</td>
<td>58.65</td>
<td>-0.62</td>
<td>55.32</td>
<td>-0.30</td>
<td>49.32</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>-0.09</td>
<td>58.33</td>
<td>-0.07</td>
<td>60.22</td>
<td>-0.12</td>
<td>58.33</td>
<td>-0.07</td>
<td>55.33</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>-0.03</td>
<td>60.21</td>
<td>-0.01</td>
<td>62.51</td>
<td>-0.05</td>
<td>62.88</td>
<td>-0.02</td>
<td>60.83</td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td>-0.17</td>
<td>58.20</td>
<td>-0.16</td>
<td>60.46</td>
<td>-0.26</td>
<td>58.84</td>
<td>-0.13</td>
<td>55.16</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 4. NLZC and the best pruning pattern**

**Table 1. IDCT complexity reduction by the proposed adaptive pruning method (APR)**
The performance of the optimized decoder by using the CBP method (OD-CBP) and the combined method (OD-APR) are shown in Table 2. The proposed OD-CBP method is applied for all I-, P-, and B-slices. If we use the proposed pruning method for both P- and B-slices, the IDCT complexity can be reduced by more from 70% to 90%. However, the subjective quality is significantly reduced. Therefore, the pruning method is applied only for B slices in order to avoid the error propagation. With the OD-CBP method, the decoder complexity is reduced by up to 7.74 % and 6.04% in average without any degradation of PSNR. The combined method (OD-APR) can reduce the decoding time by about 10% with only negligible PSNR drop.

As we discussed in Section 2, OD-CBP reduces the complexity by discarding calling IDCT and reconstructing blocks having zero CBP bit. Therefore, it does not affect the PSNR of reconstructed frames as in Table 2 (the PSNR of OD-CBP is similar to the that of JM). IDCT of blocks having zero CBP bit is discarded by OD-CBP, however, the remaining blocks have many zero coefficients. The redundant complexity related to these zero coefficients is removed by the proposed adaptive pruning method (APR). However, some non-zero coefficients are set zero so that the PSNR of reconstructed frame can be affected. As we can see in Table 2, the PSNR of OD-CBP is similar to the one of JM 17.0 while complexity of decoder is reduced. That is, the PSNR of the combined method is slightly reduced but the decoding complexity is larger than that of OD-CBP. In addition, as we can see in Fig. 5, the subjective quality of reconstructed frame with the adaptive pruning method almost remains as that by JM 17.0 with full IDCT decoding.

5. Conclusion and Future Works

In this paper, we proposed two methods that optimized the H.264/AVC decoder with reducing decoding complexity. The first method utilized the CBP information to discard calling IDCT and reconstructing blocks having CBP bit being zero. The second one adaptively pruned DCT coefficients to reduce the redundant computations of IDCT. Experimental results showed that the proposed methods can significantly reduce decoder complexity with only a negligible subjective quality drop. The proposed methods are implemented on JM. 17.0 of the H.264/AVC main profile. As a future work, we will develop a scheme that is based on DC magnitude to further reduce computational complexity of a H.264/AVC decoder.

Acknowledgement

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (No. 2012-000-5711).

References


Figure 5. Subjective quality comparison of the proposed adaptive pruning (APR) method and JM 17.0 full-IDCT decoding, The 165th frame of Bigship QP = 27: (a) Reconstructed with JM 17.0 full-IDCT decoding, (b) Reconstructed with APR(B-Slice)