Implementation of Intrapredictions, Transform, Quantization and CAVLC for H.264 Video Encoder

Bharathi S.H.¹, K. Nagabhushana Raju² and S. Ramachandran³

¹Assistant Professor in Department of Electronics and Communication Engineering, Sri Venkateshwara College of Engineering, Vidyanagar, Bangalore, India
²Associate Professor in Department Instrumentation, Sri Krishnadevaraya University, Anantapur Andrapradesh, India
³National Academy of Excellence, Bangalore, India

Abstract

Video compression algorithms operate by removing temporal and spatial redundancies in a video encoder conforming to standards such as H.264. Intra Prediction modes in H.264 improve the compression, exploiting spatial redundancy. In this work, we have implemented nine modes of intraprediction in Matlab as specified in the standard. This paper presents the algorithm for all the nine modes of intraprediction. The simulated results show better compression can be achieved in Vertical, Horizontal and Horizontal up modes than other modes for a number of pictures that have been experimented with. The compression achieved was 12 or more for these modes without sacrificing the quality of the reconstructed picture. Quality achieved was over 35 dB, indistinguishable from the original. In most of the pictures experimented with, compression improves appreciably with Intraprediction than without it up to 54%. 

Index Terms: Advanced Video coding (AVC), macro-blocks, intraprediction modes, integer transform, Quantization prediction/residual block, PSNR, and Compression.

Introduction

The Technically aligned specification of ITU-T Recommendation H.264 and ISO/IEC
MPEG 4-AVC (Advanced Video Coding), abbreviated as H.264/MPEG 4 Part 10 is an industry standard for Video Compression, the process of converting digital video into a format that takes up less capacity when it is stored or transmitted. Video compression or video coding is an essential technology for applications like Digital Television, DVD-Video, Mobile TV, Video Conferencing and Internet Video Streaming.

H.264 offers a significant performance improvement over previous video coding standards in terms of better peak signal to noise ratio and visual quality of variable block sizes for motion compensation, multiple reference frames, Integer Transform, Deblocking Filter, and Context Adaptive Variable Length Coding (CAVLC). The Intra prediction technique is one of the most important features that contribute to the success of H.264/AVC [1, 2].

The H.264 Video coding standard supports intra prediction for various block sizes. For coding the luma signal, one 16x16 macro block may be predicted as a whole or as individual 4x4 sub blocks. There are nine modes of Intra-prediction for 4x4 sub-blocks.

In this work, we have realized different blocks of AVC encoder like Integer Transform, Quantizer, Inverse Quantizer, Inverse Transform, and CAVLC in addition to nine modes of Intraprediction in Matlab.

**H.264/AVC Codec**

The block diagrams of H.264/AVC Codec are presented in Fig. 1 and 2, which includes two dataflow paths, a “forward” path and a “reconstruction” path. An input frame $F_n$ is presented for encoding. Every frame is processed in units of a Macroblock (MB) of size 16x16 pixels. Each macroblock is encoded in intra or inter mode. In both cases, a prediction macroblock $P$ is formed based on a reconstructed frame. In intra mode, $P$ is formed from samples in the current frame ‘n’ that has been previously encoded and reconstructed. The prediction $P$ is subtracted from the current macroblock to produce a residual or difference macroblock $D_n$. This is transformed and quantized to give a set of quantized transform coefficients $X$. These coefficients are reordered and entropy encoded and, the compressed bit stream is transmitted over a band-limited serial transmission channel as Network Abstraction Layer (NAL) [3].

The quantized macroblock coefficients $X$ are decoded to reconstruct a frame for encoding of further macroblocks. The coefficients $X$ are inverse quantized and inverse transformed to produce a difference macroblock $D_{n'}$. The prediction macroblock $P$ is added to $D_{n'}$ to create a reconstructed macroblock after a filter, which improves the quality of the reconstructed picture. The video sequences may also be encoded optionally using motion estimation and compensation to get a further compression of 10% or more exploiting temporal redundancy.
Decoding process consists of inverse quantization followed by inverse transform as shown in Fig. 1 as well as in the AVC decoder shown in Fig. 2. This is very similar to the forward transform and quantization. It consists of 2 stages: a scaling multiplication followed by the core inverse transform. The scaling multiplication is integrated into the inverse quantization process itself. The core inverse transform does not require multipliers and can be implemented in hardware using adders and shifters. In the decoder, the compressed bit stream NAL is entropy decoded, reordered and subjected to inverse quantization and inverse transform in order to reconstruct the picture as in the encoder. At the decoder end, only motion compensation and not motion estimation may be present.

The intra prediction reduces spatial redundancies by exploiting the spatial correlation between adjacent blocks in a given picture. Each picture is divided into 16×16 pixels MB and each MB is formed with luma components and chroma components. For luma components, the 16×16 pixels MB can be partitioned into sixteen 4×4 blocks. There are 9 prediction modes for 4×4 luma blocks and 4
prediction modes for 16×16 luma blocks. For the chroma components, there are 4 prediction modes.

In 4x4 Intra prediction modes, the values of each 4x4 block of luma samples are predicted from the neighboring pixels above or left of a 4x4 block, and nine different directional ways of performing the prediction can be selected by the encoder as illustrated in Fig. 3 and Table 1. Each prediction direction corresponds to a particular set of spatially-dependent linear combinations of previously decoded samples for use as the prediction of each input sample. Fig. 4 shows 4x4 block of pixels a, b, c ... p, belonging to a macroblock to be coded. Pixels A, B, C ...H and I, J, K, L, M are already decoded neighboring pixels used in the computation of prediction of pixels of current 4x4 block [4].

The mode 0 specifies the vertical prediction mode in which pixels (labeled a, e, i and m) are predicted from A, and the pixels (labeled b, f, j and n) are predicted from B, and so on. If Horizontal prediction is employed (mode 1), a, b, c, d are predicted by E, e, f, g, h by F etc. modes. For mode 2 (DC), all pixels (labeled a to p) are predicted by \((A+B+C+D+E+F+G+H)/8\). For mode 3 (diagonal down left), mode 4 (diagonal down right), mode 5 (vertical right), mode 6 (horizontal down), mode 7 (vertical left), and mode 8 (horizontal up), the predicted samples are formed from a weighted average of the prediction samples A to M. For example, samples a and d are respectively predicted by round \((I/4 + M/2 + A/4)\) and round \((B/4 + C/2 + D/4)\) in mode 4, also by round \((I/2 + J/2)\) and round \((J/4 + K/2 + L/4)\) in mode 8. The best prediction mode is selected for each block by minimizing the residual between the encoded block and its prediction [4].

![Figure 3: Nine Modes of 4×4 Intraprediction in H.264/AVC.](image-url)
Implementation of Intrapredictions, Transform

<table>
<thead>
<tr>
<th>Number</th>
<th>Intra 4x4 prediction mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Vertical</td>
</tr>
<tr>
<td>2</td>
<td>Horizontal</td>
</tr>
<tr>
<td>3</td>
<td>DC</td>
</tr>
<tr>
<td>4</td>
<td>Diagonal-down-left</td>
</tr>
<tr>
<td>5</td>
<td>Diagonal-down-right</td>
</tr>
<tr>
<td>6</td>
<td>Vertical-Right</td>
</tr>
<tr>
<td>7</td>
<td>Horizontal-down</td>
</tr>
<tr>
<td>8</td>
<td>Vertical-left</td>
</tr>
<tr>
<td>9</td>
<td>Horizontal-up</td>
</tr>
</tbody>
</table>

![Macroblock to be Coded](image)

**Transform, Quantization and CAVLC**

The residual macroblock \(X\) is transformed using 4x4 Integer Transform. This transform is based on the DCT with some fundamental differences [5, 6]. The 4x4 DCT of an input array \(X\) is given by

\[
Y = AXA^T = \begin{bmatrix}
    a & a & a \\
    b & c & -c & -b \\
    a & -a & a & -c \\
    c & -b & b & -c
\end{bmatrix} \begin{bmatrix}
    a & b & c \\
    a & c & -a & -b \\
    a & -c & a & b \\
    a & -b & a & -c
\end{bmatrix}
\]

where

\[
a = \frac{1}{2}, \quad b = \frac{1}{2 \cos \pi/8} \quad \text{and} \quad c = \sqrt{1/2 \cos (3\pi/8)}
\] (1)

The forward Transform is performed as

\[
Y = C_f X C_f^T \otimes E_f
\] (2)

The core Transform in Integer transform is given by

\[
W = C_f X C_f^T
\] (3)

The corresponding Inverse Transformation is given by

\[
X' = C_f^T (Y \otimes E_i) C_i
\] (4)

H.264 uses a scalar quantizer and the quantization incorporates the post and pre-
scaling matrices. The basic forward quantizer operation is as follows:

\[ Z_{ij} = \text{round} \left( \frac{Y_{ij}}{Q\text{step}} \right) \]  \hspace{1cm} (5)

Inverse quantization operation is given by

\[ Y'_{ij} = Z_{ij} \times Q\text{step} \] \hspace{1cm} (6)

The detailed steps of implementation are discussed in the previous paper and the same blocks are used for the implementation in current work.

**Context-based Adaptive Variable Length Coding**

Context based Adaptive Variable Length Coding (CAVLC) method is used to encode residual, zigzag ordered 4x4 and 2x2 blocks of transform coefficients [3]. CAVLC has several advantages as listed below:

1. After prediction, transformation and quantization blocks contain many zeros. CAVLC uses run-level coding to compactly represent strings of zeros.
2. The highest non-zero coefficients after zigzag scan are often sequences of +|-1. CAVLC numbers the high-frequency +1/-1 coefficients referred as “Trailing 1s” or “T1s” in a compact way.
3. The number of non-zero coefficients in neighboring blocks is correlated. The number of coefficients is encoded using look-up table, the choice of look-up table being dependent on the number of non-zero coefficients in neighboring blocks.
4. The magnitude of non-zero coefficients tends to be higher at the start of the reordered array and lower towards the higher frequencies. CAVLC takes advantage of this by adapting the choice of Variable Length Coding look-up table for the “level” parameter depending on recently-coded level magnitudes.

Steps in encoding CAVLC described earlier are shown in Fig. 5.

**Figure 5:** Steps in implementation of CAVLC.

**Algorithm for Various Intraprediction modes**

In the present work, all the nine modes of intra prediction have been incorporated in the Matlab implementation for H.264 Encoder. The flowchart of the entire Encoder
Implementation of Intrapredictions, Transform

implementation is presented in Fig. 6.

The input image is in TIF format. The algorithm computes PSNR and Compression for different values of Quantization steps with and without Intraprediction. The user has the choice of entering desired Quantization step for the computation. Further, it displays the Menu for the selection of Intraprediction, and the mode of Intraprediction. After the Intraprediction mode is selected, the prediction matrix P is formed from samples in the current frame that has been previously encoded. The prediction P is subtracted from the current macroblock to produce a residual macroblock ‘X’ of 4x4 size. Transformation, Quantization and their inverses are performed on ‘X’ as described in the previous section.

The Prediction matrix is updated according to the mode of Intra-prediction to process the next macroblock. PSNR is computed for the reconstructed image by computing root mean square error. CAVLC is applied to Y, C_b and C_r components as was explained in Section 3.

Very few VLSI implementations of Intraprediction, Integer Transform and Quantization [7, 8] and CAVLC [9] have been reported in the literature. In the present work, all the nine modes of Intraprediction have been successfully coded in Matlab, and therefore, we plan to implement some of these Intrapredictions on FPGAs.
Simulation Results

We have implemented H.264/AVC Encoder in Matlab for all the nine modes of Intraprediction. The simulations were performed for different image resolutions and for different QP values. For each mode, the reconstructed picture Quality (PSNR) and Compression were computed. Table 5 presents the simulated results for QP=16. The following observations may be made from the Matlab simulated results.

In most of the pictures experimented with, compression improves appreciably with Intraprediction than without it up to 54%, and without compromising on the reconstructed quality of the picture.

As QP value increases, the compression also increases. However, high value of QP (beyond 32) degrades the quality of reconstructed picture. A PSNR value greater than 30 dB is generally acceptable and, a value of 35 dB and above implies that the reconstructed picture is indistinguishable from the original.

The Compression depends on the picture contents of the original picture. From Table 5, we observe that Mode 0, Mode 1 and Mode 8 present greater compression compared to other Modes of Intraprediction for all types of pictures that have been experimented with.

All the nine modes of Intraprediction have been implemented. Of these, Vertical Mode (Mode 0), Horizontal Mode (Mode 1) and Horizontal up Mode (Mode 8) offer the highest compression (about 12 to 30), without sacrificing on the quality of the reconstructed picture (PSNR achieved is of about 34 dB to 40 dB).

As an example, reconstructed picture of Lena is presented in Fig. 7 for the three modes.
Table 5: Simulation Results of Images: PSNR and Compression Achieved with Intraprediction.

<table>
<thead>
<tr>
<th>IntrapredictionMode</th>
<th>Clock, Geneva 1024x768 pixels</th>
<th>Blue Hills 800 x 600 pixels</th>
<th>Lena 512 x 512 pixels</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR (dB)</td>
<td>Compression Achieved</td>
<td>PSNR (dB)</td>
</tr>
<tr>
<td>QP=16</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>34.0</td>
<td>11.6</td>
<td>39.0</td>
</tr>
<tr>
<td>1</td>
<td>34.1</td>
<td>12.0</td>
<td>38.5</td>
</tr>
<tr>
<td>2</td>
<td>34.0</td>
<td>7.6</td>
<td>38.9</td>
</tr>
<tr>
<td>3</td>
<td>34.2</td>
<td>10.6</td>
<td>39.8</td>
</tr>
<tr>
<td>4</td>
<td>32.6</td>
<td>8.4</td>
<td>37.0</td>
</tr>
<tr>
<td>5</td>
<td>32.8</td>
<td>8.6</td>
<td>36.1</td>
</tr>
<tr>
<td>6</td>
<td>33.1</td>
<td>9.1</td>
<td>38.2</td>
</tr>
<tr>
<td>7</td>
<td>34.5</td>
<td>11.1</td>
<td>40.0</td>
</tr>
<tr>
<td>8</td>
<td>34.6</td>
<td>11.0</td>
<td>40.3</td>
</tr>
<tr>
<td>Without Intraprediction</td>
<td>34.7</td>
<td>7.8</td>
<td>40.2</td>
</tr>
</tbody>
</table>

Figure 7: Reconstructed Lena Image Without and With Intraprediction.
Conclusions and Scope for Future work
Matlab codes have been developed for H.264 Encoder for all the nine modes of Intraprediction. The simulation results for various resolutions of pictures show better compression of about 25% in all Intraprediction Modes when compared to the Codec without Intraprediction. Of the nine modes of Intraprediction implemented, Vertical Mode, Horizontal Mode and Horizontal up Mode offer the highest compression (about 12 to 30), without sacrificing on the quality of the reconstructed picture (PSNR achieved is of about 34 dB to 40 dB). Presently, work is under progress for the implementation of these modes of Intraprediction on FPGA using the industry-standard hardware design language, Verilog.

References