International Journal of Electronics and Communication Engineering.
ISSN 0974-2166 Volume 4, Number 1 (2011), pp.105-114

http://www.irphouse.com

Implementation of Horizontal and Vertical Intra-prediction Modes for H.264 Encoder

Bharathi S.H.[1] and K. Nagabhushana Raju[2] and S. Ramachandran[3]

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

Abstract

The emerging "Advanced Video Coding" standard now known as ITU-T Recommendation H.264 and as ISO/IEC 14496 (MPEG-4) Part 10 has dominated the video coding standardization community for roughly the past three years. The new H.264/AVC standard is designed to provide a technical solution appropriate for a broad range of applications.

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 Transformation, quantization, Inverse transformation and inverse quantization for Vertical and Horizontal intraprediction in Matlab. For the reconstructed image PSNR is computed for different quantization steps. It is observed that the higher value of quantization step degrades the image quality , transformation and quantization yield compression.

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

Introduction

Broadcast television and home entertainment have been revolutionized by the advent of digital TV and DVD-video. These applications and many more were made possible by the standardization of video compression technology. The Joint Video Team (JVT)

of ISO/IEC MPEG and ITU-T VCEG are finalizing a new standard for the coding (compression) of natural video images. The new standard [1] will be known as H.264 and also MPEG-4 Part 10, "Advanced Video Coding".

H.264 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. 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, in addition two modes out of nine modes of Intra prediction are implemented 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.

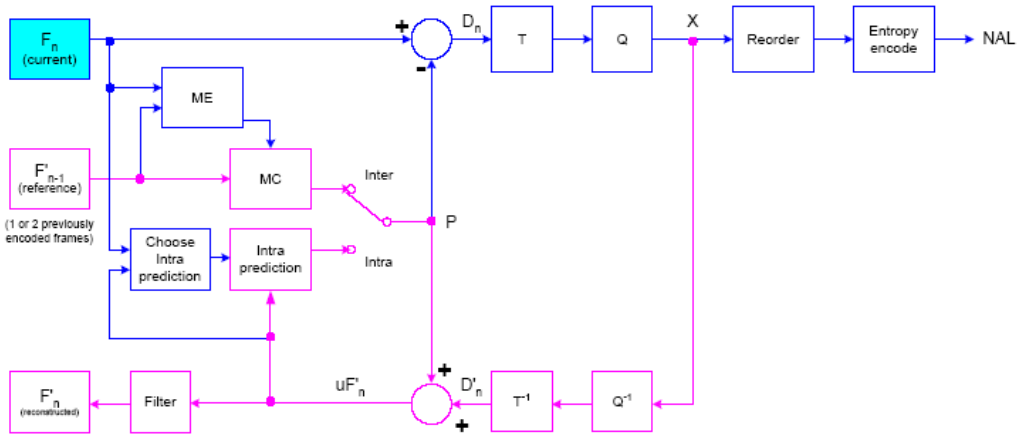


Figure 1: Block Diagram of H.264/AVC Encoder.

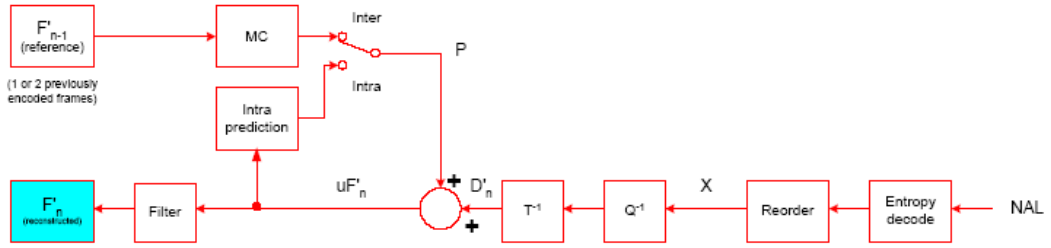


Figure 2: Block Diagram of H.264/AVC Decoder.

Intraprediction

H.264 uses the methods of predicting intra-coded macroblocks to reduce the high amount of bits coded by original input signal itself. For encoding a block or macro-block in Intra-coded mode, a prediction block is formed based on previously reconstructed blocks. The residual signal between the current block and the prediction is finally encoded. For the luma samples, the prediction block may be formed for each 4 X 4 subblock. There are nine prediction modes for 4X4 luma block as shown in Fig.3 and table 1. The direction of arrow indicates the direction of prediction. For mode 0 (vertical) and mode 1 (horizontal), the predicted samples are formed by extrapolation from upper samples [A, B, C, D] and from left samples [I, J, K, L],respectively.

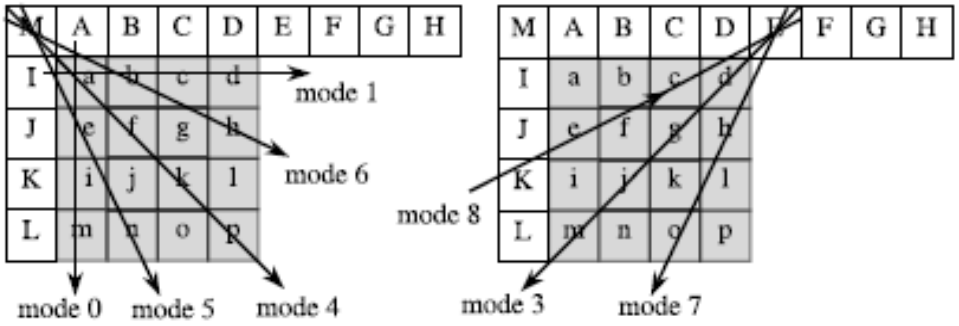


Figure 3: Nine Modes of 4×4 Intraprediction in H.264/AVC.

Table 1: Intra 4x4 Prediction Modes.

Number	Intra 4x4 prediction mode
ĭ	Vertical
I	Horizontal
ı	DC
Ŏ	Diagonal-down-left
ŏ	Diagonal-down-right
	Vertical-Right
	Horizontal-down
	Vertical-left
	Horizontal-up

The best prediction mode is selected for each block by minimizing the residual between the encoded block and its prediction [4].

Transform, Quantization Inverse Transformation and Inverse quantization

The residual macroblock X is transformed using 4x4 Integer Transform. This transform is based on the DCT with some fundamental differences [5, 6].

1. DCT is an integer transform all operations can be carried out using integer arithmetic without loss of decoding accuracy.
2. Using integer arithmetic it is possible to ensure zero mismatch between encoder and decoder.
3. The core part of the transform can be implemented using only additions and shifts.
4. Scaling multiplication is integrated in to the quantizer reducing the total number of multiplications.

The 4x 4 DCT of an input array X is given by

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

where

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

The forward Transform is performed as

$$Y = C_f X C^T_f \otimes E_f \quad (2)$$

where the forward scaling factor E_f is the matrix:

$$\begin{bmatrix} a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \\ a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \end{bmatrix}$$

The core Transform in Integer transform is given by

$$W = C_f X C^T_f \quad (3)$$

where

$$a = 1/2 \text{ and } b = \sqrt{2/5}\ .$$

The corresponding Inverse Transformation is given by

$$X' = C^T_i (Y \otimes E_i)\, C_i \quad (4)$$

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

$$Z_{ij} = \text{round}\ (Y_{ij}/Qstep) \quad (5)$$

where Y_{ij} is a coefficient of the transform, Qstep is a quantizer step size and Z_{ij} is a quantized coefficient.

The standard supports 52 values of Qstep and these are indexed by quantization parameter, QP. The values of Qstep corresponding to each QP are shown in Table 2. The quantized coefficients are obtained as follows:

$$Z_{ij} = \text{round}\ (W_{ij} \text{ x } PF/Qstep) \quad (6)$$

where PF is a^2, ab/2 or $b^2/4$ depending on the position (i, j) in Eq. 4. The factor

(PF/Qstep) is implemented in H.264 as Multiplication Factor (MF) depending on the position of the coefficient, obtained from Table 3.

Z_{ij}= round (W_{ij} x MF/2^{qbits}) (7) where MF/2^{qbits} = PF/Qstep and

qbits = 15 + floor (QP/6)

Table 2: Quantization Values: Qstep vs QP.

QP	0	1	2	3	4	5
Qstep	0.625	0.6875	0.8125	0.875	1	1.125
QP	6	7	8	9	10	11
Qstep	1.25	1.375	1.625	1.75	2	2.25
QP	12	18…..	24….	30	…..	36
Qstep	2.5	5	10	20		40
QP	……	42	……..	48		51
Qstep		80		160		224

Table 3: Multiplication Factor Values.

QP	Positions (0,0), (2,0), (2,2), (0,2) MF	Positions (1,1), (1,3), (3,1), (3,3) MF	Other positions MF
0	13107	5243	8066
1	11916	4660	7490
2	10082	4194	6554
3	9362	3647	5825
4	8192	3355	5243
5	7282	2893	4559

Example

Qp = 4, hence Qstep = 1.0

(i,j) = (0,0) , hence PF = $a^2 = 0.25$

Qbits = 1.5, hence $2^{qbits} = 32768$

$\frac{MF}{2^{qbits}} = \frac{PF}{Qstep}, hence\ MF = \frac{32768 \times 0.25}{1} = 8192$

The first 6 values of MF can be calculated as follows, depending on QP and the coefficient position (i, j):

Inverse quantization operation is given by

$$Y'_{ij} = Z_{ij} \times Qstep \quad (8)$$

The pre-scaling factor (PF) for the inverse transform is incorporated in Inverse quantization operation together with a further constant scaling factor of 64 to avoid rounding errors:

$$W'_{ij} = Z_{ij} \times Qstep \times PF \times 64 \quad (9)$$

The H.264 standard does not specify Qstep or PF directly. Instead, a parameter V is defined as Qstep x PF x 64 for $0 \leq QP \leq 5$. The inverse quantization operation is given by the equation:

$$W'_{ij} = Z_{ij} \times V_{ij} \times 2^{\text{floor}(QP/6)} \quad (10)$$

The values of V for $0 \leq QP \leq 5$ are defined as shown in Table 4.

Table 4: Values of Quantizer Step Size/re-scaling Factor V_{ij}

QP	**Positions (0,0),(2,0), (2,2),(0,2) V_{ij}**	**Positions (1,1),(1,3), (3,1),(3,3) V_{ij}**	**Other positions V_{ij}**
0	10	16	13
1	11	18	14
2	13	20	16
3	14	23	18
4	16	25	20
5	18	29	23

Example:

QP = 3, hence Qstep = 0.875 and $2^{floor(\frac{Qp}{6})} = 1$

(i,j) =(1,2), hence PF = ab =0.3162

V = (Qstep.PF.64) = $0.875 \times 0.3162 \times 65 \cong 18$

$W_{ij} = Z_{ij} \times 18 \times 1$

The values of V for QP 0–5 are defined as in Table 4.

Steps of Implementation

Step1. Input: 4X4 residual samples X

Step2. Forward "Core transformation: $Y = C_f X C^T_f \otimes E_f$

Step3. Post Scaling and Quantization: $Z = W . \frac{PF}{Qstep.2^{qbits}}$

Decoding

Step4. Re-scaling (incorporating inverse transform pre-scaling): $W^{'}_{ij}$= Z_{ij} x Qstep x PF x 64

Step5. Inverse core transform : $X^{'} = C_i^T W^{'} C_i$

Step6. Post scaling $X^{''} = round \frac{X^{'}}{64}$

Output 4X4 residual sample $X^{''}$

Simulation Results and discussions

We have implemented H.264/AVC Encoder in Matlab for Horizontal and Vertical mode of Intraprediciton. The simulations were performed for different image resolutions and for different QP values. For each mode, the reconstructed picture Quality (PSNR) is computed. Table presents the simulated results for 512 X 256 resolution image.

Table 5: Simulation Results of Horizontal and vertical Intraprediction.

Vertical Mode of Intraprediction

With Intraprediction		Without Intraprediction	
Q_Step	PSNR(dB)	Q_Step	PSNR(dB)
4	39.86	4	39.86
8	37.41	8	37.50
16	32.16	16	32.46
32	25.95	32	26.35
64	19.65	64	20.10

Horizontal Mode of Intraprediction

With Intraprediction		Without Intraprediction	
Q_Step	PSNR(dB)	Q_Step	PSNR(dB)
4	39.85	4	39.86
8	37.37	8	37.50
16	32.06	16	32.46
32	25.80	32	26.35
64	19.46	64	20.10

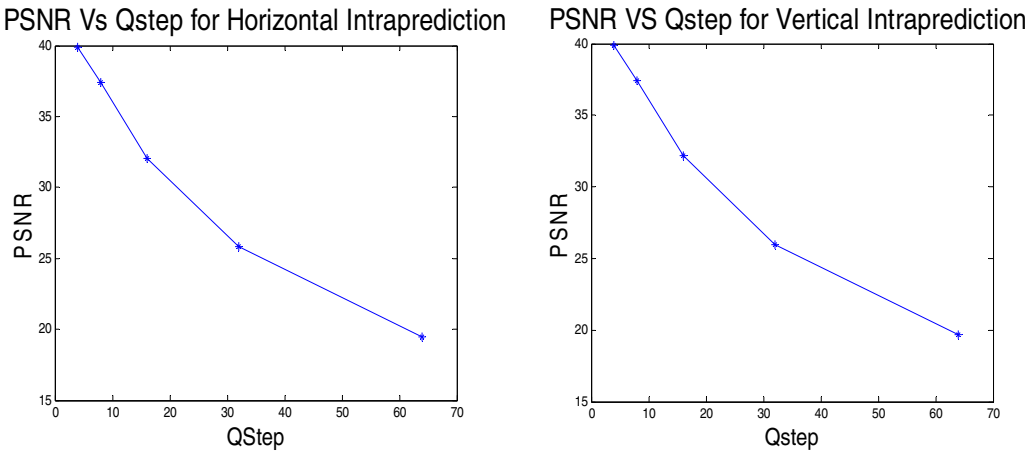
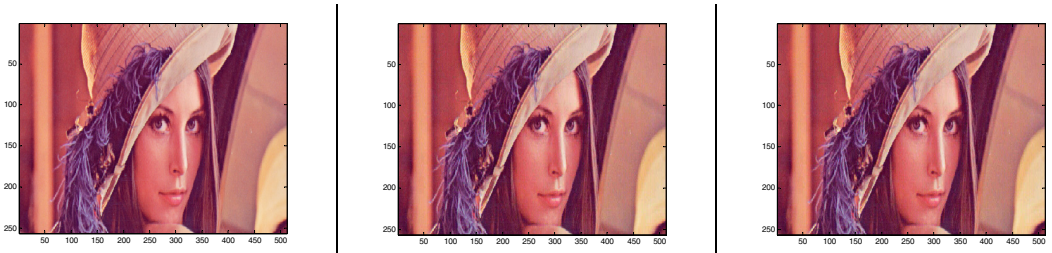


Figure 4: Graph of PSNR Vs Qstep for Horizontal and Vertical Intraprediction.



Original image 512 X 256	With Vertical Intraprediction PSNR=37.41 dB	With Horizontal Intraprediction PSNR=37.37 dB

Figure 5: Reconstructed Image in Intra prediction modes.

The following observations are made from the Matlab simulated results

1. The reconstructed image quality is poor for higher values of quantization step.
2. As quantization step value increases the PSNR value of the reconstructed image decreases.
3. Tabulated results shows the transformation and quantization process without Intra prediction for given quantization step causes compression. Hence quantization reduces the number of bits needed to store information by reducing the size of the integers representing the information in the image.
4. Vertical Intraprediction (Mode0) and Horizontal Intraprediction (Mode1) Presents almost identical results for PSNR calculations.
5. 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.

Conclusions and Scope for Future work

Matlab codes have been developed for implementation of Transformation quantization and their inverses for H.264 Encoder for Horizontal and Vertical modes of Intraprediction. The simulation results for various resolutions of pictures show better PSNR for low values of Quantization steps and transformation and Transformation and quantization process causes compression. Presently, work is under progress for the implementation of CAVLC for all nine modes of intra prediction modes to calculate the amount of compression.

References

[1] ITU-T Recommendation H.264 and ISO/IEC 14496-10 (MPEG-4) AVC, "Advanced Video Coding for Generic Audiovisual Services", Version 3: 2005.

[2] Joint Video Team, Draft ITU-T Recommendation and Final Draft International Standard of joint video specifications, ITU-T Recommendation H.264 and ISO/IEC 14496-10 AVC, March 2005.

[3] Iain E. G. Richardson, "H.264 and Video Compression", John Wiley and Sons, 2003.

[4] Feng Pan, Xiao Lin, Susanto Rahardja, Keng Pang Lim, Z. G. Li, Dajun Wu, Si Wu, "Fast mode decision algorithm for intraprediction in H.264/AVC Video coding", Circuits and Systems for Video Technology, IEEE Transactions on Volume 15, July 2005.

[5] S.Kwon,A.Tamhankar,K.R.Rao,"Overviewof H.264/MPEG-4 Part 10", IEEE Transactions on Circuits and Systems for Video Technology, 2003.

[6] Thomas Weigand, Gary J. Sullivan Gisle Bjonte gaard, and Ajay Luthra, "Over view of the H.264/AVC Video Coding Standard". IEEE Transactions on Circuits and Systems for Video Technology, July 2003.

[7] Gulistan Raja, Sadiqullah Khan, Muhammad Javed Mirza, "VLSI architecture and implementation of H.264 integer transform," The IEEE 2005 Workshop on Signal Processing Systems (SIPS'05), 2005.

[8] N. Keshaveni, S. Ramachandran, K.S. Gurumurthy, "Design and Implementation of Integer Transform and Quantization Processor for H.264 Encoder on FPGA", International Conference on Advances in Computing, Control and Telecommunication Technologies, December 2009.

[9] N. Keshaveni, S. Ramachandran, K.S. Gurumurthy, "Implementation of Context Adaptive Variable Length Coder for H.264 Video Encoder", International Journal of Recent Trends in Engineering [ISSN: 1797-9617] by the Academy Publishers, Finland, Jan. 2010.