Real-Time Software Implementation of H.264 Baseline Profile Video Encoder for Mobile and Handheld Devices

G Nageswara Rao, Prasad RSV, D Jaya Chandra and Srividya Narayanan

Emuzed India Private Limited, Bangalore, India
Email: {gnr, rsv, djayachandra, srividya}@emuzed.com

***Abstract*---The new video compression standard, H.264/MPEG-4 AVC, promises better rate-distortion performance and compression efficiency over all its predecessors. However, the computational complexity of the H.264 video codec is increased drastically because of inclusion of many new coding, error-resilience and network friendly tools and techniques, which results in it's implementation for low-end mobile and hand held devices practically difficult. This paper presents fully optimized H.264 baseline profile video encoder to show its suitability for mobile and hand-held applications. Various algorithmic and implementation techniques to optimize H.264 video encoder at low bit-rates are described. The complexity requirements for H.264 video encoder on various low-end processors are presented.**

***Index terms*---Video Coding, H.264/AVC, Baseline Profile, SIMD**

I. INTRODUCTION

H.264/AVC is the newest video coding standard [1] of the ITU-T Video Coding Experts Group and the ISO/IEC Moving Picture Experts Group. The main goals of the H.264/AVC standardization efforts are enhanced compression performance and provision of a "network-friendly" video representation addressing "conversational" (video telephony) and "non conversational" (storage, broadcast, or streaming) applications. H.264/AVC has achieved a significant improvement in rate-distortion efficiency relative to existing standards. Like previous video coding standards (MPEG-4, H263 etc), Advanced Video Coding (AVC) is also based on hybrid block-based motion compensation and transform-coding model, but it supports a lot of additional and enhanced tools. H.264 supports tree structured motion compensation with luminance block sizes of 16x16, 16x8, 8x16 and 8x8. The 8x8 sub-macro block can be further divided into partitions with block sizes of 8x4, 4x8 and 4x4. It supports motion-compensated prediction with multiple reference frames. The accuracy of displacement vectors supported by H.264 is quarter-pixel (1/4 pixel). An in-loop de-blocking filter aims at improving prediction and reducing visual artifacts. AVC adopts directional spatial prediction for intra-coding [2] to predict the pixels from the neighboring samples of already coded blocks. The conventional 8x8 floating-point discrete cosine transform is replaced by a purely integer transform at small block sizes (4x4) [2]. Transform at small block sizes helps to reduce blocking and ringing artifacts, while the integer specification prevents any mismatch between encoder and decoder. An efficient Context Adaptive Variable Length Coding [2] is adapted for H.264 baseline profile.

The field of mobile video and wireless video communication demands a better video coding technology at low bit rates and lowest computational complexities without any loss of visual quality. Although H.264 offers good efficiency [3] in terms of quality, its computational complexity, makes its implementation difficult for mobile and low power applications. Thus the real time implementations and design of fast algorithms for different coding tools is drawing lot of attention. The present work is aimed at designing different optimization techniques suitable for low-end platforms, which makes it possible to achieve real time performance for mobile video applications without any visual quality degradation. Section II explains possible algorithmic and architectural optimizations of complex modules of H.264 encoder. Results and conclusions are presented in section III.

II. OPTIMIZATION TECHNIQUES FOR H.264

This section explains the different optimization techniques in motion estimation, intra prediction process, and other major modules of H.264 encoder.

A. Intra Prediction

The standard supports intra prediction at two block sizes of 4x4 and 16x16 for luminance and at 8x8 block size for chrominance [2]. The prediction at 4x4 luma block is well suited for coding parts of a picture with significant detail. Where as, the prediction at 16x16 luma block is more suited for coding very smooth areas of a picture. Standard defines 4 modes of 16x16 block prediction, 9 modes of 4x4 block prediction and 4 modes of chroma block (8x8 block) prediction. Thus selection of best prediction mode of any macro block requires to evaluate all options and finding the one that gives the least residual signal. Thus finding best prediction mode for 16x16 luminance block requires to process about 3328 pixels (4* (16*16) + 9* (4 * 4)* 16), which is approximately equal to computation of 13 SADs (Sum of Absolute difference) for 16x16 block in motion estimation which shows that intra prediction search is equally complex as motion estimation. As both P and I frames shall use intra predicted blocks, all the frames requires evaluating best intra modes for each macro block and thus contributes considerable share in computational

complexity of encoder. Thus it makes sense to take early decisions and reduce the complexity. This section presents fast techniques to find the best intra SAD in I-frame and fast intra-inter decision process for P-frame. Intra prediction mode decision in I frame and P frame has different importance in terms of quality-complexity tradeoffs. We consider intra prediction in I frame as special case to tradeoff both coding efficiency and speed complexity as it's quality is critical for the coding efficiency of future frames.

Intra Prediction in I frame: The best mode of 16x16 block prediction are evaluated first by comparing their SADs and if it's best SAD is less than predefined threshold the remaining search at 4x4 block size is skipped. SAD computation for 16x16 blocks is done considering alternative pixels only, to minimize the computational complexity. The best prediction mode for chroma blocks is derived from best mode of corresponding 16x16 luma block.

In-place reconstruction in prediction mode search of 4x4 blocks: As the intra prediction at 4x4 block level requires the reconstructed pixels of the neighboring 4x4 blocks, after finding best mode for a 4x4 block, reconstruction is done in-place and at the same time the transformed and quantized residuals for the same block are retained. If the final prediction is at 4x4 block level, the retained quantized coefficients are coded and thus avoiding prediction, transform and reconstruction. With early exit criteria of intra prediction best mode search at 16x16 block level, the probability for re-computation would be low. The steps of intra prediction in I frame is summarized as follows.

Step1*. Compute Best intra prediction mode for 16x16 block and its SAD. If 16x16 block prediction SAD is less than a threshold exit the search for best intra mode and continue with the transform quantization and coding.*

Step 2. *Compute the best intra mode for each 4x4 block and compute the transform and quantization for the same and then reconstruct in place. Continue this step for all 4x4 blocks in scan order.*

Step 3. *Find the best intra mode of macro block by comparing 16x16 block level best mode SAD and 4x4 block level best mode SAD. If the 4x4 block level prediction is resulted as best mode, code the quantized data available in step 2. Other wise replace original pixel data and continue with prediction, transform quantization and coding for 16x16 block.*

Inter-Intra Decision: Any macro block in P frame can be predicted as INTER or INTRA, thus before it is transformed, quantized and coded, decision for INTER or INTRA need to be taken, for which distortion and bits consumption of each mode need to be evaluated. But evaluating all Intra prediction modes for the given block in P frame is costlier as frequency of occurrence of P frame in a sequence is more. It is well agreed that the probability for Intra macro blocks to occur in P Frames is very less and fast approximation of Inter-Intra decision process leads to good computational gains at the cost of negligible loss of quality. Here we proposed a fast algorithm based on evaluating one intra prediction SAD and comparing it with the inter prediction SAD resulted after motion estimation to predict the right choice of coding mode for the macro block. Here we considered DC prediction mode of 16x16 luma block as it is a better approximation for the average content in the macro block and it's simplicity of implementation for inter-intra decision. And it is empirically observed that SAD for any other intra mode would not be less than 25% of DC prediction mode SAD with high probability, and at the same time there is high probability for inter SAD to be less than $1/4^{th}$ DC SAD (about 80% of the times). The fast algorithm for inter-intra decision can be summarized as fallows,

Step 1: *If the inter SAD computed in motion estimation is less than a threshold T, decide the coding type of the macro block as INTER, and go to step 5.*

Step 2: *Compute DC prediction mode SAD for the 16x16 block. If (4 * Inter Mode SAD < DC mode SAD) Then decide the coding mode for the macro-block as INTER and go to step 5*

Step 3: *if (DC mode SAD > Th1* ***and*** *2* Inter Mode SAD < DC mode SAD) Then decide the coding mode for the macro-block as INTER and go to step 5*

Step 4: *Compute the best intra mode exhaustively as computed for I frame. Compare the SADs of best intra mode and inter mode and decide least SAD mode as the coding mode of the macro-block and go to step 5*

Step 5: *Continue with the next steps of encoding.*

The results of fast inter-intra decision algorithm are given in Table.1. The % of computational gain is calculated as percentage of macro blocks avoided in intra mode search.

Sequence	PSNR with exhaustive Inter-Intra decision	PSNR with fast inter-intra decision	% of computation gains
Foreman	33.32	33.32	84.41
News	38.21	38.19	97.28
Football	26.80	26.80	32.09
Mobile	26.18	26.16	81.58
Salesman	39.39	39.40	98.82
Suzie	38.27	38.33	92.32
Rowing	26.95	26.93	35.07
Carphone	34.09	34.12	86.72

Table 1. Avg Luma PSNR in dB and computational gains with DC SAD based fast inter-Intra decision

B. Motion Estimation

H.264 supports up to quarter pixel resolution of motion vectors and motion vectors up to 4x4 block level, as a result, the computational complexity of the motion estimation process drastically increases, which demands designing fast algorithms and implementations. With proper selection of the motion compensation tools, the motion estimation algorithm could be designed to suit the mobile applications.

Motion estimation up to 4x4 blocks is computationally complex task with insignificant coding efficiency improvements at low bit rates. It is observed that the motion estimation up to block size 8x8 is a better quality-complexity trade-off for low bit rate applications. Similarly to minimize the memory requirements and computational complexity at acceptable quality the number of reference frames can be restricted to one for mobile applications at low bit rates. This section describes the efficient implementation of fast motion estimation algorithms. Though the implementation techniques can be applied to all motion estimation techniques, our work and results given are based on Spatio-Temporal motion estimation technique [4].

1) Integer Pixel Motion Estimation

Search Strategy: The 16x16 motion vector is calculated first using a fast motion estimation algorithm based on spatio-temporal correlation of motion vectors [4]. The algorithm is designed such that search range for motion estimation dynamically gets adapted to the motion in the video. The early stopping decisions [6] for motion estimation to increase the speed are incorporated. The motion vectors for other block-sizes are calculated using a small diamond search around the 16x16 motion vector. Thereafter, R-D optimized decision is taken to select the best block-size. The SAD of 16x16 block is computed as sum of SADs of each of its 8x8 block. This allows reusing the SAD values for finding best match in sub-macro block motion estimation at 16x8, 8x16 and 8x8 block levels.

Alternate Sub-sampling with SAD value prediction: SAD calculation contributes to the most of the computational complexity in motion estimation process. To reduce this complexity, the SAD is calculated as sum of absolute differences between every alternate pixel in the corresponding blocks of reference frame and current frame. Thus, with alternate sub-sampling, the SAD_{SS} is calculated over 128 pixels in the macro block instead of 256 pixels. At the end of the calculation the SAD value is doubled to make it near equivalent to the actual value.

Sequence	Without Sub-sampling	With SAD= SADS * 2	With Weighted SAD prediction
Foreman	33.37	33.25 (49%)	33.34 (50%)
News	38.22	38.10(50%)	38.20(49%)
Football	26.79	26.65(51%)	26.74(53%)
Mobile	26.19	26.15(47%)	26.17(48%)
Salesman	39.37	39.38(47%)	39.41(46%)
Suzie	38.38	38.34(49%)	38.42(49%)
Rowing	26.95	26.82(50%)	26.93(53%)
Carphone	34.21	34.13(50%)	34.20(50%)

Table 2. Avg PSNR with Alternate Sub-sample for Integer ME (% of average number of pixels avoided in SAD computation) at QCIF 64Kpbs and 15fps encoding

The assumption here is that the SAD value computed using alternate pixels is close to the SAD value of pixels that are ignored. But when SAD_{SS} of sub-sampled block is very high then it can be observed that the SAD of alternative pixels will widely vary from each other, thus predicting the original SAD as double of that sub-sampled SAD_{SS} would not yield good comparison for different ranges of SAD_{SS} values. Thus SAD_P value prediction is modified to account the different ranges of SAD values as in equation (1),

$$SAD_P = SAD_{SS} * 2 + SAD_{SS} * k \qquad (1)$$

Where the k is a factor, which is tuned to 1/8 for 8x8, 8x16 and 16x8 blocks and is tuned to 1/16 for 16x16 blocks. Although the SAD_P value predicted from the above relation is not guaranteed to approximate to the original block SAD, it results in a value suitable for comparison of sub sampled SADs of different ranges in deciding best match. The alternative sub-sampling techniques are suitable only for architectures where SAD value is computed pixel by pixel. Thus this technique is not applied in SIMD architectures as packing and unpacking of data, counter balances the computations avoided in sub-sampling. The Table 2 shows the PSNR and complexity reductions for alternate sub-sampling with simple prediction and weighted prediction. Alternate sub-sampling along with the proposed SAD prediction technique allows using it for fractional pixel motion estimation also with negligible loss of accuracy.

2) Sub-pixel Motion Estimation

Approximation of 6-tap filter to 4-Tap filter for Motion Estimation: In H.264 half pixels are interpolated as low-pass filtering of integer pixels with 6 Taps, where [1, -5, 20, 20, -5, 1] are used as filter coefficients. In the motion estimation process 6-Tap filter can be approximated to a 4-tap filter with filter coefficients [-1, 5, 5, -1] as described in [5]. The major advantages with 4-Tap filters are computational efficiency, requires less memory bandwidth, and suitable for SIMD architectures.

4-Tap Filter implementation for SIMD architectures: The implementation of 4-Tap filtering for vertical pixels on SIMD architectures is quite difficult as efficient use of the SIMD instructions is not possible when the data is to be reordered. Here a modified vertical filtering by exploiting symmetry of filter coefficients is presented which allows the efficient implementation on SIMD architectures.

Let's say A, B, C, D are pixels consecutively placed in vertical direction. Then vertically positioned half pixels *h* between B and C are interpolated as follows.

$$h_1 = (-(A + D)>>1 + 5 *(B + C)>>1) \qquad (2)$$

$$h = \text{Clip1}\ (\ (h_1 + 2) >> 2\ ) \qquad (3)$$

The advantage with the above filtering structure is that it requires lesser unpacking instruction in SIMD architectures compared to regular structure. The averaging operation involved in Eq.2 can be computed for four pair of pixels in-

place in one operation. For example in ARM11 a 4x4 block can be loaded into four registers and registers having alternate rows of four pixels can summed and divided by 2 in one operation (using SHADD8 instruction). The quality loss with the modified structure of the 4-Tap filter is negligible. Table 3 shows PSNR differences for 4-Tap filter and modified 4-Tap filter for SIMD architectures.

Efficient quarter pixel interpolation and SAD computation: Quarter pixels are interpolated as the bilinear interpolation of adjacent integer pixel and half pixel or two half pixels. Thus merging SAD calculation of the current block with quarter pixel interpolation of the reference block will avoid the load-store for intermediate values.

Sequence	With 4-Tap filter	With SIMD 4-Tap filter	PSNR loss from 4-Tap to SIMD 4-Tap
Foreman	33.37	33.37	0.00
News	38.17	38.21	-0.04
Football	26.81	26.81	0.00
Mobile	26.17	26.18	-0.01
Salesman	39.39	39.38	0.01
Rowing	26.94	26.92	0.02
Carphone	34.21	34.24	-0.03

Table 3. Avg PSNR (dB) with 4-Tap and SIMD 4-Tap filter

C. Transform and Quantization

Quantization in the loop of transform: DCT and quantization computation for 4x4 block are merged to reduce the number of memory load and store operations.

Efficient prediction of not coded blocks and CBP calculation: Transform, quantization and coding for an 8x8 block can be skipped by predicting not coded blocks using SAD values. At another level of prediction for not coded blocks to minimize DFD (Decoded Frame Distortion) is based on cost calculated from levels and runs of quantized coefficients. Where last position of quantized non-zero coefficients is exploited to minimize the complexity involved in finding cost. The Coded block patterns for a given macro block is computed in the loop of cost calculation for prediction not coded blocks.

III. RESULTS & CONCLUSIONS

The impressive coding efficiency of H.264/AVC comes at the expense of significantly increased computational complexity compared to existing standards, which have limited the availability of cost-effective, high-performance solutions. The suitability of H264 codec for low-end applications can be proved with fully optimized codec, which meets the real time requirements of low-end applications. The paper proposes set of optimization techniques, which makes that possible on mobile platforms and respective results for each technique are presented in the corresponding sections. The complexity and PSNR results for fully optimized H.264 baseline profile encoder are presented in Table 4 to show its suitability for real time recording of video on mobile and hand-held (mobile phone / portable media player) platforms at acceptable video quality. The encoder is tested for its real time performance on Pocket PCs and Smart phones available in the market. Results are generated for QCIF resolution at 15fps, and 64kpbs. The complexity is measured on Intel wireless MMX (Bulverde) based pocket PC running at 520MHz and on ARM1136 core. The complexity of a video codec on an embedded platform is measured in mega cycles required to encode given number of frames in a second (MHz/Sec) or at a given frequency of operation number of processed frames per second (fps). Here encoded frames per second is used as complexity measure on Pocket PC and MHz/Sec with zero memory wait states is used as measure on ARM11 core. The results demonstrate achievability of high quality real time H.264/AVC recording and playback on handheld platforms.

Sequence	Complexity		PSNR in dBs
	On ARM11 core at 15 frames per second	On Intel WMMX based device	
Foreman	73.07 MHz/Sec	70.92 fps	33.39
News	46.55 MHz/Sec	62.84 fps	38.21
Football	89.67 MHz/Sec	53.17 fps	26.98
Mobile	68.58 MHz/Sec	79.46 fps	26.44
Rowing	83.59 MHz/Sec	60.42 fps	27.01
Paris	51.12 MHz/Sec	91.43 fps	34.54
Tempete	76.40 MHz/Sec	70.50 fps	27.31

Table 4. Avg Luma PSNR and Complexity for encoding QCIF resolution at bit rate of 64Kpbs on ARM1136J-S core (zero-wait states) and Intel WMMX based pocket PC

REFERENCES

[1] Joint Video Team of ITU-T and ISO/IEC JTC 1, ITU-T Rec. H.264 — ISO/IEC 14496-10 AVC, March 2003.

[2] T. Wiegand, G. J. Sullivan, G. Bjontegaard and A. Luthra, "Overview of the H.264 / AVC Video Coding Standard", IEEE Transactions on Circuit and Systems for Video Technology,VOL. 13, NO. 7, July 2003

[3] Sergio Saponara, Carolina Blanch, Kristof Denolf, Jan Bormans, "The JVT Advanced Video Coding Standard: Complexity and Performance Analysis on a Tool-by-Tool basis" Packet Video 2003, Nantes, France, April 2003

[4] K Ramkishor and S.Krishna, "Spatio-temporal Correlation Based Fast Motion Estimation Algorithm for MPEG-2", IEEE Proceedings of the 35th Asimolar conference on Signals Systems and Computers, pp. 220-224, November 2001.

[5] PSSBK Gupta and K. Ramkishor, "Novel Algorithm to Reduce Complexity of Quarter Pixel Motion Estimation" Visual Communication and Image Processing (VCIP), California, Jan'2004.

[6] K. Ramkishor, PSSBK Gupta, T. S. Raghu, and K. Suman, "Algorithmic Optimizations for Software-only MPEG-2 Encoding", IEEE Transactions on Consumer Electronics, Vol. 50, No. 1, Feb'2004.