SELECTION OF VARIABLE BLOCK SIZES IN H.264

A. Ahmad, N. Khan, S. Masud, M.A. Maud

Department of Computer Science, Lahore University of Management Sciences,
Sector-U, D.H.A., Lahore 54792, Pakistan
Email: smasud@lums.edu.pk

ABSTRACT

This paper aims at selecting an efficient variable block size mode in H.264 video coding standard for better compression performance. This standard allows video frames to be partitioned into variable block sizes such that blocks containing high detailed motion are represented using small bock sizes and the rest using large block sizes. New techniques for intelligent selection of the variable block sizes have been developed to reduce the computational complexity without sacrificing the quality of coder. The proposed schemes are based on the motion vector cost and previous frame information. An improvement in the encoding time with negligible impact on the subjective and the quantitative performance has been achieved. A comparison of the proposed techniques for various test sequences is also provided.

1. INTRODUCTION

H.264 is a video compression standard being jointly developed by ITU-T Video Coding Experts Group and ISO/IEC Motion Picture Expert Group. The main goal of this standardization effort is enhanced compression performance and provision of a network-friendly packet-based video representation. The applications primarily include video communications over the Internet and mobile wireless channels. Various robust and error-resilience features have therefore been incorporated in these recommendations. Variable block size is a feature of H.264 that allows varying the block size according to the locally changing characteristics so that an additional compression gain can be achieved.

H.264 uses a block based motion vector search algorithm. Several approaches have been presented in the literature to determine the best choice of motion vectors. Vaisey and Gersho discussed techniques [1] in which the size of the block in motion estimation is varied according to the local detail of the image using quad-tree implementation. Kim and Lee described [2] a rate-distortion (RD) constrained approach for the hierarchical quad-tree variable block size (VBS) motion estimation and displaced frame difference (DFD) coding. Another approach has been proposed that uses an RD based motion estimation method employing a hierarchical VBS structure [3]. However, the previous published work does not take into consideration the block size information of the previous and the current frame in encoding motion vectors of the streams. Since contiguous frames are spatially and temporally related, significant saving in motion estimation cost is possible by using variable block sizes and exploiting this correlation.

This paper presents new intelligent techniques for the selection of the block size modes taking into consideration the modes calculated in the previous and current frames, the motion vector cost and the average SNR of the frames. The schemes presented here are comparable in coding performance to the reference scheme yet the computational complexity is reduced.

The rest of the paper is organized as follows: Section 2 gives a brief discussion of the variable block size features of H.264. The existing mode selection scheme in the employed H.264 encoder and the measures for performance evaluation are included in section 3. Section 4 discusses the new approaches incorporated in the encoder in order to reduce its complexity for selection of the block sizes intelligently. This is followed by results and conclusion.

2. VARIABLE BLOCK SIZE

Unlike the previous video coding standards such as MPEG 1 and H.263, the H.264 standard allows variable block sizes to be used in motion estimation. This is in addition to using only 16x6 or 8x8 block sizes. It permits seven different block sizes: they are 16x16, 16x8, 8x16, 8x8, 8x4, 4x8 and 4x4. A separate motion vector can be transmitted for each block size. Thus for a 16x16 macroblock, a maximum of 16 motion vectors can be sent, one for each 4x4 block. The use of variable block sizes helps in adapting the data to the changing characteristics of the video. The blocks with higher motion detail can be coded using smaller block size that helps in improving the prediction by taking into consideration fine details in motion. Similarly, the blocks with less motion detail can be encoded using larger block sizes [4].

The use of variable block sizes for motion compensation results in a decrease in the bitstream size as compared to fixed block sizes. It can be easily observed that as the block size is reduced there is an increase in the number of encoded bits since motion vector for each block has to be transmitted. The VBS scheme hence produces the minimum number of bits for encoding all kinds of video sequences albeit at the cost of additional computational complexity.

3. MODE SELECTION IN JVT MODEL AND EVALUATION MEASURES

For the purpose of evaluation, a public reference encoder JVT Model (JM) v 6.1e [5] is used as a starting point. The software was tested on a system based on Intel Pentium III processor with 128 MB RAM. The encoder was compiled using a commercial

C++ compiler and was run under the Microsoft Windows 2000 Professional operating system. Intel VTune Performance Analyzer has been used to get the profiling results [6].

In the reference implementation, the variable block size modes have been divided into two types: macroblock level modes and 8x8 chroma level modes. The macroblock level modes consist of 16x16, 16x8 and 8x16 block sizes whereas the 8x8 chroma modes consist of 8x8, 8x4, 4x8 and 4x4 block sizes. In order to select the best mode for a block, motion vector cost is calculated for all the modes. The mode that gives the minimum cost is selected as the best mode and used for encoding. The coder supports two cost calculation criteria. One is the Motion Vector (MV) based cost and the other is the RD based cost. The MV based cost is calculated by using a lambda factor defined as:

MV_COST = (WEIGHTED_COST (f,mvbits[((cx)<<(s))-px]+ mvbits[((cy)<<(s))-py]))

where

f = lambda factor,
cx, cy = candidate x and y position for motion estimation,
s = motion vector shift
px, py = predicted x and y position for motion estimation,

and

WEIGHTED_COST = returns the cost for the bits using the lambda factor

The RD-cost scheme, on the other hand, takes into consideration the distortion factor and the rate of the compressed stream. The distortion is computed by calculating the SNR of the block and the rate is calculated by taking into consideration the length of the stream after the last stage of encoding The proposed mode selection techniques use the MV_COST for deciding the best mode for encoding as explained in section 4.

3.1 Comparison Criteria

Analysis has been done on five standard video sequences in QCIF (176x144) format each representing a different class of motion. These include 'news', 'foreman', 'coastguard', 'trevor' and 'silent'. The first 100 frames of each of the above mentioned sequences at a frame rate of 30 fps have been used for this purpose. These sequences have been selected on the basis of variety of motions that they contain. The 'news' sequence contains motion in the foreground as well as in the background. The 'foreman' sequence contains camera movement. The 'coastguard' is a relatively fast motion sequence with camera movement. The 'trevor' sequence contains six insets, each containing different motion sequences. Finally, the 'silent' sequence contains a static background. The sequences were encoded using new VBS mode selection schemes described later in section 4.

The reference encoder consists of the seven variable block sizes with one reference frame option. The quality of the encoded stream is determined through the average SNR values for the luminance and the chrominance components. The subjective video quality was also observed for all the proposed schemes. The complexity is determined by the average time it takes for the encoding of the frames at a frame rate of 30 fps. The efficacy of the scheme is evaluated by the total size of the encoded stream compared to the reference implementation. Profiling results are used to ascertain the increase in the complexity of the software as a result of the implementation of the new approaches.

4. PROPOSED TECHNIQUES FOR MODE SELECTION

The following new techniques have been developed to incorporate intelligent mode selection in the H.264 encoder.

4.1 SNR Based Selection

The first approach developed to reduce the computational complexity defines a Signal-to-Noise Ratio (SNR) based criteria that prevents calculation of the VBS modes every frame. A threshold average SNR value of 32 dB for the luminance component of the frames is defined. The modes are calculated for the first frame and stored in memory. The second frame uses the corresponding modes in the first frame for coding of a block. Average SNR of the second frame is calculated. If the average SNR of the second frame does not fall below the threshold average SNR value then the same modes are used for the third frame and so on. Otherwise, if the average SNR falls below the threshold value then it indicates that the modes are not suitable for encoding and have to be re-calculated for the next frame. The modes in this situation are determined as in the reference implementation described in section 3.

The results show an average decrease from 31% to 34% in the encoding time of the codec for all the chosen sequences. There is a 7% increase in the bitstream size in 'Coastguard' sequence. For other sequences, the addition in bitstream size is between 16% to 20%. This can be attributed to the degree of motion and the camera movement involved in these sequences. The subjective quality of the streams does not change in all cases. The average SNR values compared to reference implementation results do not differ and they are in the range of 33 dB to 36 dB.

4.2 Adaptive Threshold Cost Based Selection

This algorithm uses an adaptive threshold cost based selection criteria to decide the best mode for the current block. In this method, the modes of the previous frames along with their motion vector cost are stored in memory. The motion vector cost of the same mode in the previous frame is used as the threshold cost. Since it is unlikely that the cost calculated in the current frame is exactly equal to the cost calculated in the previous frame, therefore a range of 10-30% more or less of the previous cost is used to evaluate the current macroblock cost, i.e.

If

(prev_cost(100-n)%<MV_COST(mode)< prev_cost*(100+n)%)*

where $n \in \{10, 20, 30\}$

then the VBS modes are not calculated for the block. Otherwise the modes are recalculated by checking all the possible modes as

Table 1: Total bits (KBits) for Adaptive Threshold Cost Based Selection

	10%	20%	30%
News	255.23	257.52	274.07
Foreman	377.71	377.71	377.71
Coastguard	747.54	761.84	771.85
Trevor	430.74	437.42	452.56
Silent	272.56	279.05	284.24

Table 2: Encoding time when using Adaptive Threshold Cost Based Selection

	10%	20%	30%
News	168.3	158.0	148.0
Foreman	177.1	178.6	178.2
Coastguard	173.6	160.7	149.9
Trevor	167.7	159.4	148.3
Silent	166.7	155.3	153.1

in the reference encoder. Table 1 shows that in the adaptive threshold based mode selection scheme, as the percentage threshold range increases, the total bits for the sequences also increase. The increase in total encoded bits is from 3% to 9% as compared to the reference implementation for 30% threshold range. The less the threshold range the more modes will be checked and the encoding time will generally increase. This is apparent from Table 2. The percentage decrease in time for 30% threshold range as compared to the reference implementation is 36% to 40% for all test sequences. The reduction in encoding time implies reduced computational complexity and improved real-time performance.

4.3 3D-Recursive Search Scheme Based Selection

The 3D Recursive Search (3D-RS) based algorithm has been used in the motion estimation of the video sequences [7]. It makes use of the information in the previous and the current frame to calculate motion vectors. The VBS algorithm proposed here uses the 3D-RS algorithm for the selection of suitable mode for motion estimation in video sequences. The algorithm employs the modes calculated for the current and the previous frame to determine the mode for the current macroblock. Modes for the first column and the first row of the macroblock are calculated by exhaustively testing all the modes. For the rest of the macroblock there are three conditions. They are:

a) If the macroblock is located in the middle then the modes of the macroblocks which are to the left, up-left and up-right to the current macroblock in the current frame, and the macroblock that are two down and one right to the current macroblock in the previous frame are used. The cost for current macroblock is calculated using these four modes and the mode with the minimum cost is selected for encoding. So,

min(MV_COST(mode of A), MV_COST(mode of B) MV_COST(mode of C) MV_COST(mode of D))

Thus, four modes are checked for each macroblock at maximum instead of checking all seven modes.

b) If the macroblock is on the right edge then the macroblocks to the left and up left to the current macroblock are used to evaluate the cost.

c) If the macroblock is located on the bottom edge then the macroblocks to the left, up-left and up right of the current macroblock are used to evaluate the cost.

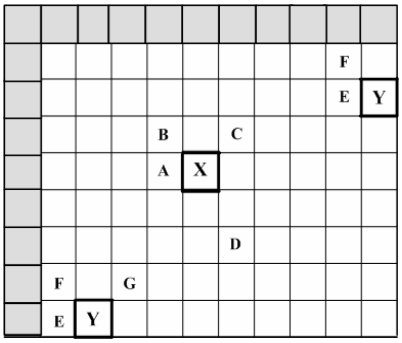

Figure 1: Macroblocks used for selection of the best mode for a QCIF Sequence

Figure 1 is a representation of the macroblocks selected to determine the suitable mode for the current macroblock. Modes for the first row and column, shown by gray blocks in figure 1, are selected by thoroughly checking all the modes and selecting the mode with the minimum cost as in the reference implementation. For macroblock X, the modes of macroblocks A, B, C and D are used and the mode that has the least cost is chosen. The modes used for A, B and C are the ones that have been calculated in the current frame. The mode used for D is as calculated in the previous frame. Similarly, if the macroblock is on the right edge then only the modes for the macroblock E and F are used. If the macroblock is at the bottom edge then the macroblock E, F and G are used to determine the best mode.

The results for this algorithm, discussed in section 5, show that the encoding time is similar to other proposed techniques whereas the increase in bitstream size is less and the subjective quality of the compressed stream remains unchanged. The percentage improvement in the encoding time over reference implementation is from 27% to 32%. There is a 5% to 18% increase in the bitstream sizes over the reference implementation in all sequences. The average SNR for the luminance component for the sequences is also comparable to the reference implementation.

5. RESULTS

The encoding time for all the sequences is depicted in figure 2. The threshold range used for the adaptive threshold cost based selection is 30%. The results show that there is a substantial decrease of over 30% in the encoding time of the sequences when compared to the reference implementation of the variable block size mode selection. The 3D-RS based mechanism is the

most computationally intensive among the proposed techniques as it has to normally calculate cost for four modes as compared to adaptive threshold based algorithm and SNR based algorithm which only calculate the cost of one mode if the condition is satisfied.

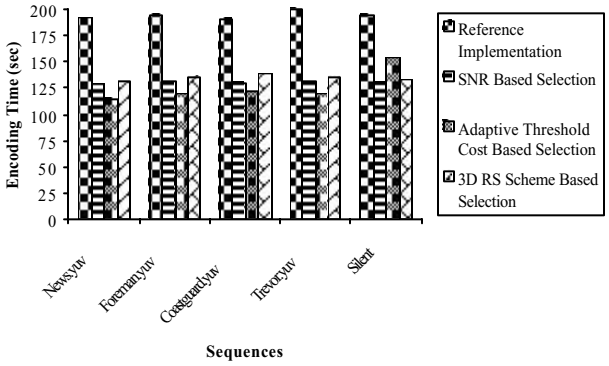

Figure 2: Comparison of the encoding time of the sequences

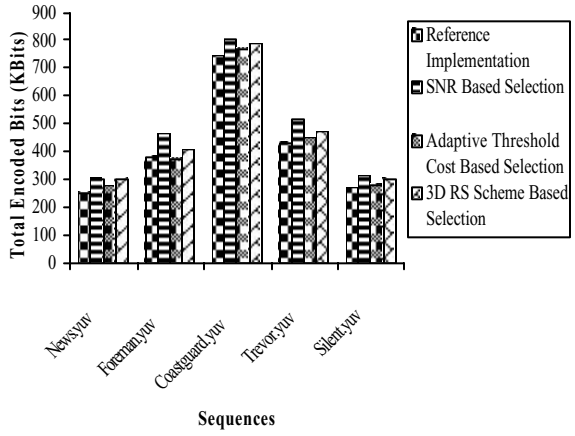

Figure 3: Comparison of total encoded bits of the sequences

Figure 3 represents the total encoded bits for each sequence using the proposed schemes. Since the reference implementation is an exhaustive search for the best mode, therefore the number of bits for each sequence is the least. The 3DRS and the adaptive threshold based selection afford similar compression performance.

Figure 4 illustrates the comparison of the average SNR of the sequences. As can be seen, the results of the proposed algorithms are comparable to that of the reference implementation. Among the proposed algorithms 3DRS based mode selection provides acceptable SNR for most of the sequences. The profiling results obtained from VTune show that the percentage decrease in the total time taken by the main module of the encoder through the proposed mode selection techniques is around 70%. Thus these methods facilitate real-time implementation of H.264 codec.

6. CONCLUSION

This paper presents algorithms for enhancing the performance of variable block size mode selection in H.264. The new approaches for the selection of the best mode result in significant reduction in the computational complexity at the cost of a slight degradation in the compression performance of the codec. The subjective quality of the compressed stream, however, does not change. The average SNR for the luminance component remains same as the reference algorithm through the use of proposed mode selection schemes. The adaptive threshold based mode selection algorithm provides least computational load with improvement in the encoding time.

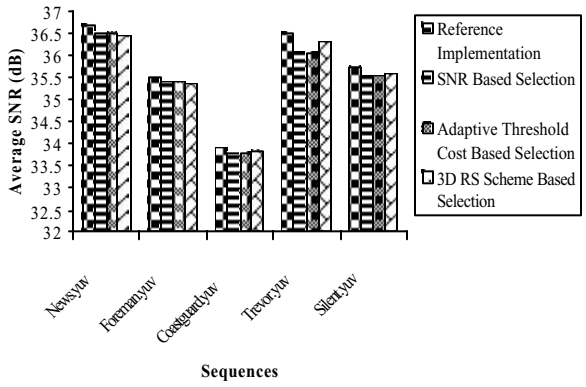

Figure 4: Comparison of Average SNR of encoded sequences

7. REFERENCE

[1] D. Jacques Vaisey and Allen Gersho, "Variable Block-Size Image Coding", Department of Electrical and Computer Engineering University of California Santa Barkma, CA 93106

[2] Jong Won Kim and Sang Uk Lee , "Video Coding with R-D Constrained Hierarchical Variable Block Size(VBS) Motion Estimation" presented at SPIE's Visual Communication and Image Processing '95, Taipei, Taiwan, May 23-26, 1995.

[3] Sang Uk Lee and Jong Won Kim , "Variable Block Size Techniques for Motion Sequence Coding" , Signal Processing Lab., Dept. of Control & Instrumentation Eng., Seoul Nat'l University, Shinlim-Dong, Kwanak-Ku, Seoul 151-742, KOREA.

[4] "Emerging H.26L Standard: Overview and TMS320C64x Digital Media Platform Implementation", www.ubvideo.com October 21, 2003.

[5] http://bs.hhi.de/%7Esuehring/tml/download/jm61e.zip October 21, 2003.

[6] Intel VTune Performance Analyzer http://www.intel.com/software/products/vtune/index.htm October 21, 2003.

[7] S.Oliveru, L.Albani, G. de Haan, "A Low-Complexity Recursive Search Motion Estimation Algorithm For H.263 Video Coding", Philips Research Laboratories Monza (MI) and Eindhoven.