J. Software Engineering & Applications, 2010, 3: 472-476
doi:10.4236/jsea.2010.35053 Published Online May 2010 (http://www.SciRP.org/journal/jsea)

A Novel Efficient Mode Selection Approach for H.264

Lu Lu, Wei Zhou

School of Computer Science & Engineering, South China University of Technology, Guangzhou, China.
Email: lul@scut.edu.cn

Received March 11th, 2010; revised April 2nd, 2010; accepted April 3rd, 2010.

ABSTRACT

H.264 video coding standard introduces motion estimation with multiple block sizes to achieve a considerably higher coding efficiency than other video coding algorithms. However, this comes at the greatly increased computing complexity at the encoder. In this paper, a method is proposed to eliminate some redundant coding modes that contribute very little coding gain. The simulation results show that the algorithm can remarkably decrease the complexity at the encoder while keeping satisfying coding efficiency.

Keywords: *Video Coding, H.264, Mode Selection*

1. Introduction

The JVT (Joint Video Team) introduced a number of advanced features in H.264 or MPEG-4 AVC. These improvements achieve significant gains in encoder and decoder performances [1-3]. One of the new features is multi-mode selection, which is the subject of this paper. In the H.264 coding algorithm, blockmatching motion estimation is an essential part of the encoder to reduce the temporal redundancy between frames. H.264 supports motion estimation and compensation using different block sizes ranging from 16 × 16 to 4 × 4 luminance samples, which is shown in **Figure 1**, with many options between the two. The luminance component of each macroblock can be split by four ways: 16 × 16, 16 × 8, 8 × 16 and 8 × 8. Each of the submacroblock partitions is called a macroblock partition. If the 8 × 8 mode is chosen, each of 8x8 macroblock partitions within the macroblock can be further split by four ways: 8 × 8, 8 × 4, 4 × 8 or 4 × 4, which are called macroblock sub-partitions. These partitions and subpartitions give rise to a lager number of possible combinations within each macroblock.[1]

H.264 standard uses computationally intensive Lagrangian rate-distortion (*RD*) optimization to choose the best block size for a macroblock. The general equation of Lagrangian *RD* optimization is given as:

$$J_{\mathrm{mod}\,e} = D + \lambda_{\mathrm{mod}\,e} \bullet R \tag{1}$$

where J_{mode} is the rate-distortion cost (*RD* cost) and J_{mode} is the Lagrangian multiplier; *D* is the distortion measurement between original macroblock and reconstructed macroblock located in the previous coded frame, and *R* reflects the number of bits associated with choosing the mode and macroblock quantizer value, Qp, including the bits for the macroblock header, the motion vector(s) and all the DCT residue blocks [4,5].

The computational complexity required by motion estimation, however, increases linearly with the number of used block types because block matching needs to be performed for each of them. In JVT reference software JM75C[6], it adopts full search method for each block type and selects the optimal block type as the final coding mode based on the *RD* cost function. Though it provides the best coding efficiency, the computational complexity is obviously much too high. In order to reduce the intensive computational requirement, Andy Cbang etc. proposed fast multi-block motion estimation [7]. They adopt an approach of early termination by skipping searching for mode 16 × 8 and mode 8 × 16, if the performance of mode 16 × 16 is “good enough”, otherwise all coding modes will be performed. This method only considers three coding modes which are 16 × 16, 16 × 8 and 8 × 16 inter coding modes. Another approach, proposed by Andy C. Yu, is based on estimating block detail complexity [8]. It is an effective way judging by his simulation results, but there is more a critical factor, texture direction, which he does not think about but also can be useful to significantly improve coding efficiency.

In this paper, we propose a effective method to eliminate some redundant coding modes in mode selection.

[1] This paper is supported by Guangdong Technology Project (2009B010800048) and Guangzhou Technology Major Project.

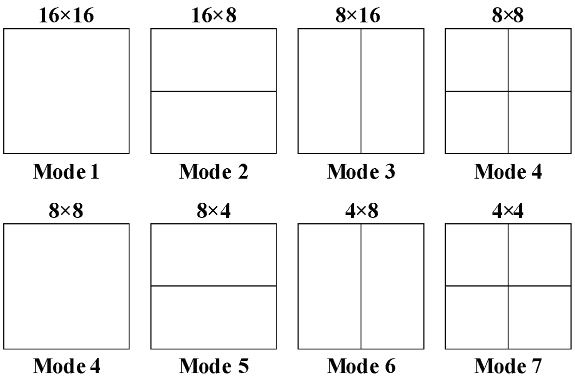

Figure 1. Inter-prediction modes

The paper will be organized as follows. The proposed algorithm will be described in detail in Section 2. Section 3 shows the simulation and the results. Finally, a conclusion will be given in Section 4.

2. Proposed Algorithm

2.1 Block details

Table 1 shows the observations on how selected modes relate sequence characteristics.

The choice of partition size has a significant impact on compression performance. In general, according to **Table 1**, large partition sizes are appropriate for homogeneous areas of the frame and small partition sizes may be beneficial for detailed areas.

We derive an approach based on summing the total energy of the AC coefficients to estimate the block detail. The AC coefficients can be obtained from the DCT coefficients of each block. The definition is:

$$E_{AC} = \sum_{u=1}^{M-1} \sum_{v=1}^{N-1} (F\ (u,v))^2 \tag{2}$$

From (2), E_{AC}, the total energy of the AC components of an $M \times N$ block is the sum of all the DCT coefficients, $F(u,v)$, except for the DC component, $u = 0$ and $v = 0$.

$$F(u,v) = c(u)c(v)$$
$$\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \cos[\frac{(2x+1)u\pi}{16}] \cos[\frac{(2y+1)v\pi}{16}] \tag{3}$$

where,

$$c(u), c(v) = \begin{cases} \sqrt{\frac{1}{M}}, \sqrt{\frac{1}{N}} & for\ u,v=0 \\ \sqrt{\frac{2}{M}}, \sqrt{\frac{2}{N}} & for\ u,v \neq 0 \end{cases} \tag{4}$$

According to the energy conservation principle, the total energy of an $M \times N$ block is equal to the accumulated energy of its DCT coefficients. Thus, (3) can be further simplified as

$$E_{AC} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} (f^2(x,y)) - \frac{1}{MN} [\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y)]^2 \tag{5}$$

where the first term is the total energy of the image intensities within an $M \times N$ block, and the second term represents the mean square intensity. Equation (5) clearly shows that the energy of the AC components of a macroblock can be represented by the variance.

Evaluating the maximum sum of the AC components is the next target. By definition, the largest variance is obtained from the block comprising checkerboard pattern in which every adjacent pixel is the permissible maximum and minimum value. Thus, $E_{\max}$, the maximum sum of AC components of an M×N block is

$$\begin{aligned} E_{\max} &= MN \frac{f_{\max}^2(x,y) + f_{\min}^2(x,y)}{2} \\ &- \frac{MN}{4} [f_{\max}(x,y) + f_{\min}(x,y)]^2 \end{aligned} \tag{6}$$

Note that $E_{\max}$ can be calculated in advance. Then the criterion to assess the complexity of a macroblock detail is

$$r_d = \frac{\ln(E_{AC})}{\ln(E_{\max})} \tag{7}$$

In total, 7 different block sizes are recommended by H.264 for P-frames, namely, 16 × 16, 16 × 8, 8 × 16, 8 × 8, 8 × 4, 4 × 8, 4 × 4 as well as SKIP, and other two INTRA prediction modes, I4MB and I16MB. However, in our complexity measurement, there are only 3 categories, which are denoted as MD16 category, MD8 category, and MD4 category, respectively.

The proposed algorithm provides a recursive way to decide the complexity of each macroblock. Firstly, a macroblock of 16 × 16 pixels is examined with the first

Table 1. Selected modes for different sequences

Sequence	Skip	16×16	16×8	8×16	8×8	Intra16	Intra4
Container	75.8	10.4	3.5	2.7	7.3	0.3	0.0
Foreman	23.7	39.9	39.9	7.3	7.6	7.3	9.3
Bus	3.5	22.0	12.1	14.4	40.5	1.0	5.5
Mobile	4.5	31.3	7.1	6.1	6.1	49.7	0.3

IPPP*, *5 reference frames*, *CABAC*, *CIF Format

piecewise equation in (7). An LDB category is given if it is recognized as being a homogenous macroblock. Otherwise, the macroblock is decomposed into 4 blocks of 8 × 8 pixels. Note that an 8 × 8 block is recognized as high-detailed if it satisfies two conditions: 1) the RB in (7) is greater than 0.7, and it is decomposed into four 4 × 4 block, and 2) one of its four decomposed 4 × 4 blocks is highdetailed as well. If an 8 × 8 block satisfies the first condition but not the second, it is still recognized as low-detailed. After checking all the 8 × 8 blocks, an MDB category is given to a macroblock which possesses more than two high-detailed blocks, otherwise the HDB category is assigned. **Table 2** displays the relationship between the three categories in the proposed algorithm and the 9 inter-frame prediction modes. It is observed that the LDB category covers the least number of prediction modes, whereas the HDB category contains all the available modes. The table further indicates that the higher detailed the macroblocks are, the more prediction modes the proposed algorithm has to check.

The function of the natural logarithm is to linearize both $E_{\max}$ and E_{AC} such that the range of r_d can be uniformly split into 10 subgroups. In our evaluation, a macroblock that has the r_d >0.7, is considered to be a high-detailed block.

2.2 Object Movement

More than one object is contained in a macroblock and is moving in different directions. This included objects moving over a background with different velocity. For example, in **Figure 2** the object is moving against a static background. In this case, the current block should be divided into two 8 × 16 sub-blocks whereas sub-block 0 should have a zero motion vector and sub-block 1 should have a motion vector such that the cost function can be minimized.

2.3 Texture Regions

When the edge of texture aligned perfectly with the sensor boundaries at a particular time instant, the texture edge is clear and sharp. We will describe this texture as having "integer-pixel location". When the texture undergoes an integer-pixel translational motion, the texture will look exactly the same in the two consecutive frames except that one is a translation to another. And the moved texture can be predicted perfectly by integer-pixel motion estimation.

If the edges of texture have a half-pixel offset relative to the senor, the edges may be blurred as shown in **Figure 3(b)** and said to have "half-pixel location". The original zero-pixel-wide (sharp) edge now becomes one pixel-wide (blurred). The pixel at the blurred edges may have only half the intensity of the original one, which can lead to difficulty in motion estimation.

Similarly, if the edges have a quarter-pixel offset it may be blurred as shown in **Figure 3(c)**. We will describe this texture as having "quarter-pixel location". The zero-pixel-wide (sharp) object edge becomes one pixel-wide (blurred). The pixels at the blurred edges may have 3/4 or 1/4 of the intensity. The use of sub-pixel motion estimation algorithm, like half-pixel or quarter-pixel estimation, uses interpolation to predict the sub-pixel shift of texture relative to the sampling grid.

Table 2. Block categories and corresponding modes

Detail Level	Enabled Modes
LDB	16×16
MDB	16×16, 16×8, 8×16, 8×8
HDB	8×8, 8×4, 4×8, 4×4

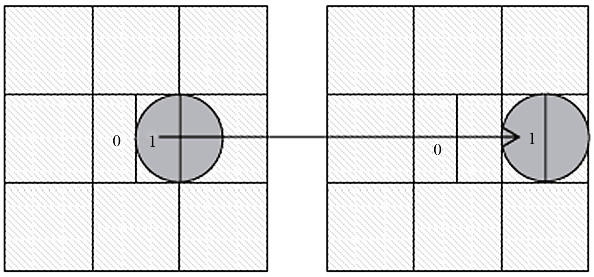

Figure 2. Example of an object moving on a static background

Different type of texture (integer, half or quarter) has a different response to fractional motion estimation. For example, the texture in **Figure 3(b)** (half) can be predicted perfectly by the texture in **Figure 3(a)** (integer) using half-pixel motion estimation but not vice verse. Since it is possible for a macroblock to contain more than one kind of texture, using only one integer, half or quarter pixel motion vector will not be sufficient to describe the texture content. For example, a macroblock may contain two 8 × 16 sub-blocks where sub-block 0 contains "half-pixel" texture and sub-block 1 contains "integer-pixel" texture. In this case, the current macroblock should be divided into two 8 × 16 sub-blocks in which half-pixel motion vector should be used for sub-block 0 and integer-pixel motion vector for sub-block 1.

2.4 Algorithm

Former results [9] show that, often, about 70% of the macroblocks will choose mode 1 (16 × 16) as their final block type. In the proposed algorithm it determines the macroblock detail-level and analysis the information obtained from 8 × 8 block size ME to predict the mode 1 macroblock in advance, if possible, the optimal motion vector. If the macroblock is predicted to be mode 1 macroblock, searching will be stopped immediately.

As a result, computation can be saved for mode 2 and mode 3 block size ME and in some situation mode 1 as well. Three decisions are set up in handling different

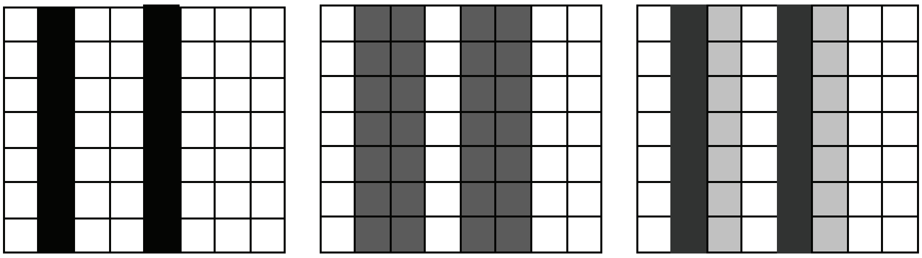

Figure 3. (a) Integer-pixel texture; (b) Half-pixel texture; (c) Quarter-pixel texture

video area-general area, slow moving area and fast moving area.

Step1: If r_d<0.3 then

- Select 16 × 16 as the only enabled mode (LDB)

Else if 0.3<r_d<0.7 then

- Disable 8 × 8, 4 × 8, 4 × 4 (MDB)

Else if If r_d>0.7 then

- Enable all of the modes (HDB)

Defining MV0, MV1, MV2, MV3 be the motion vector of 8 × 8 subblock of the current macroblock. Two conditions are checked:

Step2:

C1: If MV0=MV1=MV2=MV3 then

- choose mode 1 (16 × 16) as final block type
- no ME will be further performed
- 16 × 16 MV = 8 × 8 MV0

C2: If three subblock MV are the same AND the forth unequal MV only differ by one quarter pixel (1/4)

distance

then

- choose mode 1 (16 × 16) as final block type
- no ME will be further performed
- 16 × 16 MV = dominated 8 × 8 MV

C3: If collocate MB in previous frame is model AND {MV0, MV1, MV2, MV3} < 4 (*i.e.* one integer pixel distance) AND MV0, MV1, MV2, MV3 has the same direction then

- choose mode 1 (16 × 16) as final block type
- 8 point local search around MV = {0, 0}

C4: If all magnitude of 8 × 8 MVx >= 3 integer distance OR all magnitude of 8 × 8 MVy >= 3 integer distance

- choose mode 1 (16 × 16) as final block type
- local search for surrounding 24 points of MV0

The reason for all the 8 × 8 motion vector having the same direction in decision C3 can be illustrated using **Figure 4(a)**. Suppose a macroblock is undergoing small rotational motion as shown in **Figure 4(a)**. The motion vector at the left size of macroblock will be downward and the right side will be upward. As a result, there is a high potential for the current macroblock segmented vertically even the magnitude of motion vector is very small.

Figure 5 shows the performance of C1 + C2 + C3 + C4. We can see the hit rate increase for the fast panning part of foreman sequence which is much closer to the optimal one.

3. Simulation Results

The proposed algorithm was implemented in the reference JVT software.JM75C. We have tested our proposed method over a series of testing sequence with different resolution.

In this paper, two QCIF (176 × 144) sequences, "Foreman" and "Stefan" are selected to show the result. In the simulation, the sequences are encoded at 30 fps with qp = 10 to 20 with step size of two. The PSNR and bitrate comparison between proposed algorithm and full search is shown in **Table 3**.

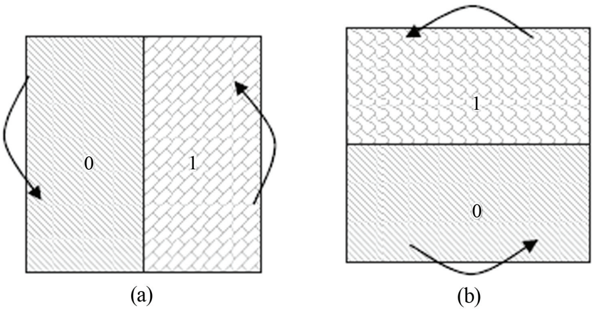

Figure 4. Example of rotational motion in Macroblock that cause segmentation (a) vertical segmentation; (b) horizontal segmentation

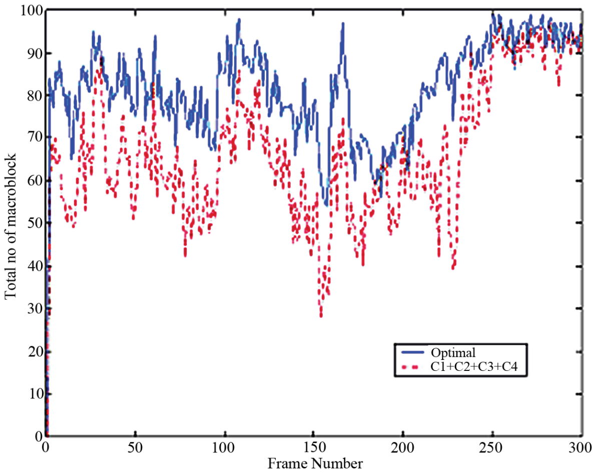

Figure 5. Performance using Decision C1 + C2 + C3 +C4 using foreman QCIF sequence

Table 3. PSNR and Bitrate Comparison between the proposed algorithm and FS with QP = 10 to 20; (a) Stefan QCIF; (b) Foreman QCIF

Stefan QCIF							ForemanQCIF								
	FMFME		Full Search					FMFME		Full Search					
QP	Psnr (dB)	BR (kbits)	Psnr (dB)	BR (kbits)	Gain (dB)	BR Gain	QP	Psnr (dB)	BR (kbits)	Psnr (dB)	BR (kbits)	Gain (dB)	BR Gain
10	49.48	2602.9	49.48	2600.0	0	-0.11%	10	49.69	1457.1	49.69	1455.0	0	-0.15%
12	47.64	2203.3	47.64	2201.3	0	-0.09%	12	47.97	1149.2	47.97	1146.8	0	-0.21%
14	46.13	1891.5	46.14	1889.5	-0.01	-0.10%	14	46.5	923.26	46.5	921.76	0	-0.16%
16	44.48	1611.5	44.48	1608.4	0	-0.19%	16	44.89	732.14	44.9	729.57	-0.01	-0.35%
18	42.58	1323.5	42.58	1321.5	0	-0.15%	18	43.11	552.33	43.12	550.69	-0.01	-0.30%
20	40.89	1095.2	40.89	1092.2	0	-0.27%	20	41.52	422.31	41.53	419.7	-0.01	-0.62%
22	39.3	903.06	39.3	900.83	0	-0.25%	22	40.03	328.54	40.03	326.17	0	-0.73%
24	37.36	707.5	37.36	705.61	0	-0.27%	24	38.32	241.99	38.33	238.92	-0.01	-1.28%
26	35.65	557.72	35.66	555.8	-0.01	-0.35%	26	36.83	180.03	36.85	178.54	-0.02	-0.83%
28	33.95	432.34	33.96	430.4	-0.01	-0.45%	28	35.48	136.92	35.49	135.35	-0.01	-1.16%
30	32.06	322.08	32.07	320.79	-0.01	-0.40%	30	33.99	103.02	34	101.24	-0.01	-1.76%
32	30.34	238.65	30.34	236.84	0	-0.76%	32	32.57	77.65	32.58	76.58	-0.01	-1.40%
34	28.79	177.98	28.8	177.61	-0.01	-0.21%	34	31.3	60.67	31.34	59.62	-0.04	-1.76%
36	27.13	127.37	27.13	127.04	0	-0.26%	36	29.96	45.86	30.03	45.43	-0.07	-0.95%
38	25.66	94.1	25.68	93.35	-0.02	-0.80%	38	28.63	35.55	28.71	35.47	-0.08	-0.23%
40	24.33	70.58	24.36	70.35	-0.03	-0.50%	40	27.49	28.63	27.53	28.3	-0.04	-1.17%
				Average	-0.00625	-0.32%					Average	-0.02	-0.82%

(a) (b)

The complexity is shown in **Table 3**. The proposed algorithm can reduce computational cost by 58% on average (equivalent complexity of performing motion estimation on 1.7 block types instead of 4 block types) with negligibly small PSNR degradation (0.013dB) and slight-increase in bit rate (0.57%).

4. Conclusions

In this paper, we propose a method to eliminate some redundant coding modes, which speeds up the process of multi-mode selection. The simulation results show that the algorithm can remarkably decrease the complexity at the encoder while keeping satisfying coding efficiency.

REFERENCES

[1] "Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG: Draft Text of Final Draft International Standard for Advanced Video Coding," *H.* 264|*ISO/IEC* 14496-10 *AVC*, ITU-T.

[2] M. Ghanbari, "Standard Codecs: Image Compression to Advanced Video Coding," IEE Publishing, 2002.

[3] E. G. Iain and Richardson, "H.264 and MPEG-4 Video Compression," Wiley, 2003.

[4] F. S. Yan, "Fast mode selection based on texture analysis and local motion activity in H.264/AVC," 2004 *International Conference of Communications, Circuits and Systems*, Chengdu, Vol. 1, 27-29 June 2004, pp. 539-542.

[5] G. W. Teng, Z. Y. Zhang, Y. J. Zhang and W. J. Zhang, "Fast Mode Decision Algorithm in Inter Pictures Based on H. 264/ AVC," *Journal of Optoelectronics·Laser*, Vol. 16, No. 7, July 2005, pp. 866-870.

[6] "JVT Reference Software JM75C". http://bs.hhi.de/~suehring/tm

[7] A. Chang, O. C. Au and Y. M. Yeung, "A Novel Approach to Fast Multi-block Motion Estimation for H.264 Video Coding," *Proceedings* 2003 *International Conference on Multimedia and Expo*, Maryland, Vol. 1, 6-9 July 2003, pp. 539-542.

[8] A. C. Yu, "Efficient Block-size Selection Algorithm for Inter-Frame Coding in H.264/MPEG-4 AVC," 2004 *IEEE International Conference on Acoustics, Speech and Signal Processing*, Montreal, Vol. 3, 17-21 May 2004, pp.69-72.

[9] Y. S. Cui, D. G. Duan and Z. L. Deng, "Fast Motion Estimation Algorithm on H.264," *Journal of Liaoning Institute of Technology*, Vol. 24, No. 5, October 2004, pp. 12-15.