Guoping Qiu Kin Man Lam
Hitoshi Kiya Xiang-Yang Xue
C.-C. Jay Kuo Michael S. Lew (Eds.)

LNCS 6297

Advances in Multimedia Information Processing – PCM 2010

11th Pacific Rim Conference on Multimedia
Shanghai, China, September 2010
Proceedings, Part I

1 Part I

Springer

Guoping Qiu Kin Man Lam
Hitoshi Kiya Xiang-Yang Xue
C.-C. Jay Kuo Michael S. Lew (Eds.)

LNCS 6298

Advances in Multimedia Information Processing – PCM 2010

11th Pacific Rim Conference on Multimedia
Shanghai, China, September 2010
Proceedings, Part II

2 Part II

Springer

Efficient Coding of Motion Vector Predictor Using Phased-in Code

Ji-Hee Moon and Yo-Sung Ho

Gwangju Institute of Science and Technology (GIST)
261 Cheomdan-gwagiro, Buk-gu, Gwangju, 500-712, Korea
{jhmoon,hoyo}@gist.ac.kr

Abstract. The H.264/AVC video coding standard performs inter prediction using variable block sizes to improve coding efficiency. Since we predict the motion in the homogeneous region as well as in the non-homogeneous region accurately using variable block sizes, we can reduce residual data effectively. However, each motion vector should be transmitted to the decoder. In low bit rate environments, the motion vector information takes approximately 40% of the total bitstream. Thus, motion vector competition was proposed to reduce the amount of motion information. Since the size of the motion vector difference is reduced by motion vector competition, it requires only a small number of bits for motion vector information. However, we need to send the corresponding index of the best motion vector predictor to the decoder. In this paper, we propose a new codeword table based on the phased-in code to encode the index of the motion vector predictor efficiently. Experimental results show that the proposed algorithm reduces the average bit rate by 7.24% for similar PSNR values, and it improves the average image quality by 0.36dB at similar bit rates.

Keywords: KTA, motion vector competition, phased-in code.

1 Introduction

The H.264/AVC standard is the latest video codec developed by the joint video team (JVT). For higher compression efficiency, H.264/AVC has adopted several powerful coding techniques, such as variable block-size macroblock modes, multiple reference frames, integer discrete cosine transform (DCT), and efficient entropy coding techniques. For inter prediction, the combination of variable block sizes, SKIP mode prediction, and 1/4-pel motion estimation are allowed to improve the motion compensation efficiency. This generates a compression gain with an increase of bits about motion information. In low bit rate environments, the proportion of bits for motion information takes approximately 40% of the total bit rate [1].

After the H.264/AVC standard was established, the video coding experts group (VCEG) has attempted new challenges: (1) doubling the compression gain compared to H.264/AVC under equivalent quality, (2) reducing coding complexity, and (3) enhancing error resilience. In the 26th VCEG meeting, it was decided to establish the KTA (key technology area) software, which gathers all efficient tools for video

G. Qiu et al. (Eds.): PCM 2010, Part II, LNCS 6298, pp. 340–349, 2010.

coding. The joint collaborative team (JCT) has been established to develop the high efficiency video coding (HEVC) standard. Therefore, the KTA techniques have greater importance than before.

The current version of the KTA software that is based on JM 11.0 contains adaptive interpolation filter (AIF), adaptive quantization matrix selection (AQMS), and mode dependent directional transform (MDDT). Especially, the motion vector competition (MVComp) scheme is regarded as an effective coding tool to reduce motion vector data. In the MVComp scheme, the combination of spatial and temporal neighboring motion vectors generates motion vector predictors and the optimal motion vector predictor is selected by rate distortion optimization (RDO). Since MVComp reduces the size of the motion vector difference, it requires only a small number of bits for motion vector information. However, the MVComp scheme requires additional information to indicate which motion vector predictor is used. This additional data reduces coding efficiency of MVComp [2].

In order to overcome this problem, we propose an efficient coding method to represent the index of the optimal motion vector predictor. We assign codewords of variable lengths to the index of the motion vector predictor based on the distribution of the optimal motion vector predictor. Since the codeword table is based on the phased-in code, we can reduce the average length of the codeword for the index of the motion vector predictor.

This paper is organized as follows. We introduce the conventional motion vector coding and an overview of competition-based motion vector coding in Section 2. We propose a new codeword table for encoding the index of the optimal motion vector predictor in Section 3. In Section 4, experimental results are presented and we draw conclusions in Section 5.

2 Overview of Competition-Based Motion Vector Coding

2.1 Motion Vector Coding

H.264/AVC supports block-based motion compensation with various block sizes to improve coding efficiency. The sizes of block are 16×16, 16×8, 8×16, and P8×8. The P8×8 can be divided into 8×8, 8×4, 4×8, and 4×4 in each 8×8 block. Inter prediction using motion compensation of variable block sizes predicts not only the motion of the homogeneous region but also the motion of the non-homogeneous region accurately.

After motion estimation, we transmit residual data by subtracting a prediction macroblock from the current macroblock. The decoder reconstructs the current macroblock by adding the residual to the prediction macroblock. The decoder uses the motion vector to create a prediction macroblock. The motion vector indicates the offset from the coordinates in the current block to the coordinates in the reference block. Therefore, the bitstream that represents the inter macroblock includes residual data and the motion vector.

In order to represent the motion vector, differential pulse code modulation (DPCM) is applied. The motion vector predictor is similar to the current motion vector which is determined using motion estimation. Thus, the motion vector difference between the

current motion vector and the motion vector predictor is very small compared to the current motion vector. Therefore, in terms of coding efficiency, it is better than encoding the original value of the current motion vector. The motion vector difference D_{mv} is defined by

$$D_{mv} = mv - pmv \tag{1}$$

where mv is the current motion vector and pmv is the motion vector predictor.

Motion vectors for neighboring macroblocks are highly correlated and the motion vector predictor is obtained from vectors of nearby and previously coded macroblocks. The method of forming the motion vector predictor depends on the partition size and availability of nearby motion vectors. Figure 1 shows the location of neighboring macroblocks to predict the motion vector of the current macroblock.

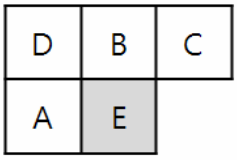


Fig. 1. Current and neighboring macroblocks

Let E be the current macroblock, macroblock partition, or sub-macroblock partition. Let A be partition or sub-partition immediately to the left of E. Let B be the partition or sub-partition immediately above E and let C be the partition or sub-macroblock partition above and to the right of E.

The motion vector predictor of the current macroblock E is determined by median of the motion vectors for partition A, B, and C.

$$pmv = median(mv_A, mv_B, mv_C) \tag{2}$$

If one or more of the previously encoded blocks shown in Fig 1 are not available, the choice of the motion vector predictor is modified accordingly. At the decoder, the motion vector predictor is formed in the same way and added to the decoded motion vector difference. The motion vector difference is encoded by the singed Exponential Golomb code [3].

2.2 Motion Vector Competition

In the MVComp scheme, the encoder organizes the candidate set of all possible distinct motion vector predictors for the current block. Since there are a variety of motion vector predictors, the size of the motion vector difference is reduced and the bits for motion information are also decreased. Therefore, the MVComp scheme achieves improvement of coding efficiency.

The candidate set includes spatial and temporal properties of neighboring motion vectors. Figure 2 shows the candidate motion vector predictor set of the current macroblock. The available spatial motion vector predictors are neighboring motion

vectors ($mv_{a,}$ $mv_{b,}$ mv_c) and the H.264/AVC median motion vector (mv_{median}). The temporal motion vector predictor is motion vector at the same macroblock position in the reference frame ($mv_{collocated}$). The additional motion vector predictor is zero motion vector (mv_{zero}). These motion vector predictors are given for inter mode and SKIP mode. In particular, the candidate set of SKIP mode adopts the extended spatial motion vector predictor. The extended spatial motion vector predictor is slightly different from the median motion vector. The extended spatial motion vector predictor depends on the availability of neighboring motion vectors. If three neighboring vectors are available, it returns the median value of these. However, if only mv_a is available, the extended spatial motion vector predictor becomes mv_a. In the same manner, the extended spatial motion vector predictor becomes mv_b or mv_c according to the availability of neighboring motion vectors. If all three neighboring vectors are not available, the extended spatial motion vector predictor is zero [4].

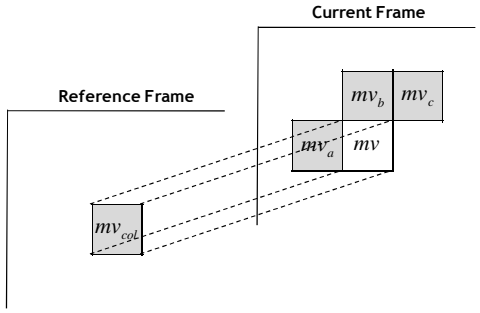


Fig. 2. Candidate motion vector predictor set

In order to select the optimal motion vector predictor among the candidate motion vector predictors, we define the rate distortion optimization function. The optimal motion vector predictor is selected by minimum rate distortion.

$$J = D + \lambda R \tag{3}$$

where D is the distortion between the original block and the motion compensated block. And λ is the weighted value which depends on the quantization parameter. R is the bit rate which is defined as follows.

$$R = R_m + R_r + R_{mv} + R_{mm} + R_o \tag{4}$$

where R_m is the rate for the macroblock mode and R_r is the rate of residual data. R_{mv} is the rate of the motion vector difference corresponding to the motion vector predictor and R_{mm} is the rate for the index of the optimal motion vector predictor. R_o is the rate of the other components: slice header, coded block pattern, stuffing bits, delta quantization, and reference index. The distortion is computed in the spatial or transform domain and the rate components are estimated or really encoded in the exact number of bits.

For the SKIP mode, the rate distortion optimization is defined by

$$J_{skip} = D_{skip} + \lambda(R_m + R_{mm}) \tag{5}$$

where D_{SKIP} is the distortion generated by the SKIP mode. R_m is signaling of the SKIP mode. R_{mm} represents the number of bits for the index of the motion vector predictor. In the MVComp scheme, the fixed length code is used to encode the index of the optimal motion vector predictor. Since the size of the motion vector difference is reduced by MVComp, it requires only a small number of bits for encoding motion vector information. However, it has a disadvantage to encode the corresponding index of the optimal motion vector predictor for decoding.

3 Proposed Algorithm

3.1 Distribution of Motion Vector Predictor

For a given inter-coded macroblock, KTA chooses the optimal motion vector predictor from six different candidates of motion vector predictors: mv_a, mv_b, mv_c, mv_{median}, $mv_{collocated}$, and mv_{zero}. KTA also provides the additional motion vector predictor for the SKIP mode. In order to consider the distribution of the best motion vector predictor, we examined the selected motion vector predictor for five different sequences (Akiyo, Foreman, Paris, Stefan and Mobile). Figure 3 shows the average distribution of the best motion vector predictor for the SKIP mode and the inter mode. The occurrence probability of the median motion vector which is selected as the optimal motion vector predictor is higher than other motion vector predictors for both macroblock types. Also, the collocated motion vector predictor is frequently selected as the optimal motion vector predictor for the inter mode.

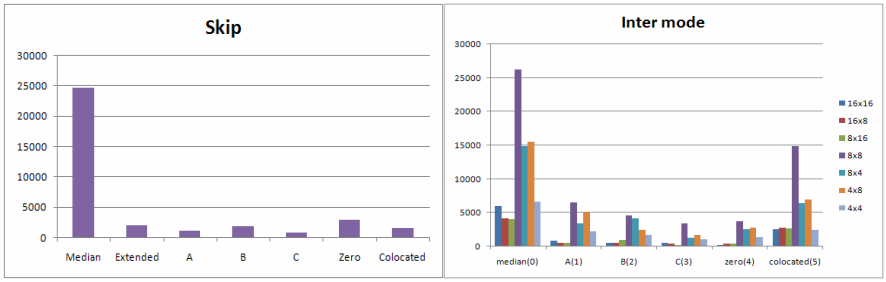


Fig. 3. Distribution of the best motion vector predictor for the SKIP mode and the inter mode

As shown in Fig. 3, the distribution of the best motion vector predictor is not uniform. Therefore, it is more effective to assign the shorter codeword to symbols that occur more frequently than symbols that occur less frequently. We design a new index table that has codewords with two different lengths. The median motion vector predictor and the collocated motion vector predictor have the shorter codeword compared to the others.

3.2 Motion Vector Predictor Index Coding Using Phased-in Code

The conventional KTA uses the fixed length code for encoding the index of the motion vector predictor. The codeword length of the index is determined by Eq. (6).

$$Length = (maxmode/2) + (maxmode \bmod 2) \tag{6}$$

The inter mode uses six different motion vector predictors. The three bits are used for representing the index of the optimal motion vector predictor. In the SKIP mode, there are seven different motion vector predictors. The four bits are assigned to the index of the optimal motion vector predictor. Although the SKIP mode has seven different motion vector predictors, the four bits are determined due to Eq. (6).

In this paper, we propose the efficient codeword table based on the phased-in code [5]. We consider the distribution of the motion vector predictor as well as the phased-in code. The phased-in code has a codeword composed of two kinds of length. In general, Data where symbols have the equal probabilities cannot be compressed by the fixed length code. Even if symbols have the equal probability, the phased-in code assigns the different length codeword to symbols. Thus, we can obtain compression effect using the phased-in code. Since the phased-in code is a prefix code, it has no ambiguity for decoding the index of the motion vector predictor. Another advantage of the phased-in code is easy to encode symbols and decode bitstream.

The design method of codeword table for the inter mode is as follows [6].

1) In the inter mode, there are six different motion vector predictors. The number of symbols to be encoded is six. n is equal to six.
2) We compute an integer m. m+1 is the minimum number of bits for representing each symbol. In order to construct the codeword for the index of the motion vector predictor, we need at least three bits. Thus, m is equal to two.
3) We compute the p and P. p is n-2^m and P is 2^m-p. The sum of $2p$ and P is equal to the number of whole symbols (equal to n). In the inter mode, p is two (=6-2^2) and P is two (=2^2-2).
4) Given the integer index of the motion vector predictor to be encoded, if it is less than P, the encoder constructs an m bits codeword with the index value. For example, if the index value is smaller than two, the index value is represented using two bits.
5) We construct an m+1 bits codeword for remaining symbols. The codeword has value from 2^{m+1}-$2p$ to 2^{m+1}-1. In the inter mode, the index values from two to five are converted into the index values from four to seven. The mapped values are assigned to three bits codeword.

From the above process, we can design the efficient codeword table for the SKIP mode. The SKIP mode has the seven different motion vector predictors. m is equal to two and p is three and P is equal to one. If the symbol is smaller than one, the symbol is represented using two bits codeword. For reminders, symbols are converted into two to seven and are represented using three bits codeword.

Table 1 shows the codeword of the motion vector predictor for the inter mode. Table 1 is based on the phased-in code. The index number zero and five indicate the median motion vector predictor and the collocated motion vector predictor,

respectively. These indices are converted into values from zero to one and the mapped values are assigned to two bits codeword.

Table 1. The codeword of the motion vector predictor for the inter mode

Motion vector predictor	Index	Conventional codeword	Proposed codeword
mv_{median}	0	000	00
mv_a	1	001	100
mv_b	2	010	101
mv_c	3	011	110
mv_{zero}	4	100	111
$mv_{collocated}$	5	101	01

Table 2 shows the codeword for encoding the index of the motion vector predictor. In the SKIP mode, the median motion vector predictor occurs frequently. The index that is equal to zero means the median motion vector predictor. We assign two bits codeword to encode the index number zero. The remaining symbols are converted into values from two to seven and these symbols are encoded using three bits codeword.

Table 2. The codeword of the motion vector predictor for the SKIP mode

Motion vector predictor	Index	Conventional codeword	Proposed codeword
mv_{median}	0	0000	00
$mv_{extended}$	1	0001	010
mv_a	2	0010	011
mv_b	3	0011	100
mv_c	4	0100	101
mv_{zero}	5	0101	110
$mv_{collocated}$	6	0110	111

The conventional method to encode the index of the motion vector predictor uses three bits codeword for each index in the inter mode. The SKIP mode uses the four bits for each index. We cannot obtain the performance of MVComp due to the additional bits for the index of the selected motion vector predictor.

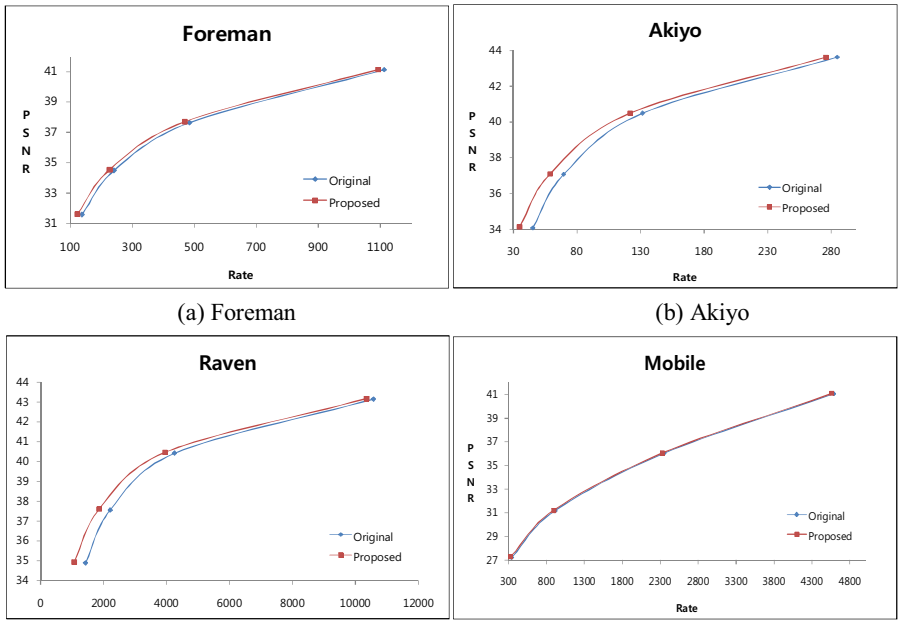
However, the proposed algorithm based on the distribution of the best motion vector predictor assigns two or three bits codeword to the index that occurs frequently. In the inter mode, we assign two bits codeword to the indices of the median motion vector predictor and the collocated motion vector predictor. The remaining symbols are represented using three bits codeword. In the SKIP mode, we assign two bits and three bits codewords to the median motion vector predictor and other motion vector predictors, respectively. We can reduce the average length of bits for encoding the index. Thus, we can improve the performance of MVComp.

4 Experimental Results and Analysis

In order to evaluate the performance of the proposed algorithm, we used KTA 2.6 software [7]. We encoded 100 frames from the four test video sequences in the CIF (352×288) format and 50 frames for three test video sequences in the HD (1280×720) format. The detailed encoding parameters for the experiment are summarized in Table 3. When *MVcompetition* is equal to two, it indicates that we can adjust the number of motion vector predictors and the type of motion vector predictor.

Table 3. Encoding parameters

ProfileIDC	66 (baseline profile)
QP	22, 27, 32, 37
SearchRange	16
SymbolMode	0 (CAVLC)
Frame Structure	IPPP..P
MVCompetition	2
Predictors_SKIP	11111110
Predictors_MVp	10111110



(a) Foreman (b) Akiyo

(c) Raven (d) Mobile

Fig. 4. Rate distortion curves

Figure 4 illustrates the rate distortion curves for several sequences. We can confirm that the proposed algorithm achieve the improvement of performance, especially in

low bit rate environments. The Bjøntegaard delta peak signal-to-noise ratio (BDPSNR) and the Bjøntegaard delta bit rate (BDBR) were used to evaluate the performance of the proposed algorithm [8]. The negative value of BDBR and the positive value of BDPSNR indicate the improvement of encoding performance. The results for several test sequences are shown in Table 4. We achieved the average bit savings by 7.24% and PSNR improvement by 0.36dB.

Table 4. Comparison of BDPSNR and BDBR

Test Sequence	QP	KTA		Proposed		BDPSNR (dB)	BDBR (%)
		PSNR (dB)	Bit rate (kbits/s)	PSNR (dB)	Bit rate (kbits/s)		
Foreman	22	41.13	1112.17	41.14	1093.29	0.28	-5.80
	27	37.64	484.76	37.7	469.11		
	32	34.5	241.17	34.54	226.73		
	37	31.59	137.84	31.62	123.01		
Akiyo	22	43.63	284.71	43.63	275.81	0.82	-12.15
	27	40.5	131.63	40.5	121.68		
	32	37.08	69.5	37.1	58.8		
	37	34.08	45.18	34.14	34.84		
Mobile	22	41.08	4587.66	41.1	4562.02	0.11	-2.02
	27	36.05	2350.19	36.06	2325.79		
	32	31.19	912.49	31.21	894.28		
	37	27.23	340.5	27.28	326.81		
Paris	22	40.75	1417.67	40.75	1399.53	0.20	-3.39
	27	36.81	758.9	36.81	742.32		
	32	32.73	371.55	32.74	356.94		
	37	29.12	186.05	29.1	171.64		
Raven	22	43.15	10571.76	43.18	10354.1	0.66	-13.19
	27	40.42	4250.4	40.46	3944.11		
	32	37.55	2206.15	37.6	1858.94		
	37	34.89	1427.01	34.94	1065.71		
Bigship	22	40.49	21754.41	40.52	21551.56	0.33	-8.53
	27	36.84	6203.86	36.87	5975.37		
	32	33.81	2371.3	33.84	2104.59		
	37	31.37	1336.58	31.4	1042.93		
Crew	22	42.21	19594.95	42.22	19404.26	0.20	-5.58
	27	39.47	6244.21	39.49	6058.25		
	32	37.16	2916.28	37.18	2707.83		
	37	34.95	1753.92	34.99	1533.1		
Average						0.36	-7.24

From Fig. 4, we can confirm that the performance of simple sequences such as Akiyo and Raven is higher than that of complicate sequences. In the simple sequence, the SKIP mode and the 16×16 inter mode are selected as the best macroblock mode. However, the complicate sequence chooses the P8×8 mode as the best macroblock mode. When we use the proposed algorithm, the amount of reduced bits of the P8×8 mode is larger than those of the SKIP mode and the 16×16 mode. Since the selection

of the P8×8 mode is increased, the video quality is improved. However, the disadvantage is that we need to send more bits about information which represents the P8×8 mode. Although we reduce the bits for representing the index of the motion vector predictor, the bits that indicates the P8×8 mode is increased. From this reason, we cannot obtain the improvement of the performance in complicate sequence such as Mobile.

5 Conclusions

In this paper, we proposed a new index table for encoding the motion vector predictor in KTA. We considered the distribution of the selected motion vector predictors in both the SKIP mode and the inter mode and we developed the index table of the motion vector predictor. These tables are based on the phased-in code. We can assign less bits to the median motion vector predictor and the collocated motion vector predictor in the inter mode. In the SKIP mode, the median motion vector predictor has codeword of two bits. Experimental results showed that the proposed algorithm reduced the bit rate by 7.24% under equivalent PSNR values.

Acknowledgement

This research was supported by the MKE (Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the NIPA (National IT Industry Promotion Agency) (NIPA-2010-(C1090-1011-0003)).

References

1. Wiegand, T., Sullivan, G., Bjontegaard, G., Luthra, A.: Overview of the H.264/AVC video coding standard. IEEE Transactions on Circuits and Systems for Video Technology 13(7), 560–576 (2003)
2. Competition-Based Scheme for Vector Selection and Coding, document VCEG-AC06.doc, ITU-T SG16/Q6 (July 2006)
3. Marpe, D., Wiegand, T., Sullivan, G.: The H.264/MPEG4 Advanced Video Coding Standard and Its Application. IEEE Communication Magazine 44(8) (August 2006)
4. Laroche, G., Jung, J., Pesquet-Popescu, B.: Competition Based Prediction for SKIP Mode Motion Vector Using Macroblock Classification for the H.264 JM KTA software. In: Blanc-Talon, J., Philips, W., Popescu, D., Scheunders, P. (eds.) ACIVS 2007. LNCS, vol. 4678, pp. 789–799. Springer, Heidelberg (2007)
5. Jin, H., Seung-Hwan, K., Yo-Sung, H.: New CAVLC Design for Lossless Intra Coding. In: International conference on Image Processing, November 2009, pp. 637–640 (2009)
6. Salomon, D.: Variable-length Codes for Data Compression. Springer, Heidelberg (2007)
7. KTA Software Version 2.6. r1, `http://iphome.hhi.de//suehring/tml/download/KTA/jm11.0kta2.6.r1`
8. Improvement of the BD-PSNR Model, document VCEG-AI11.doc, ITU-T SG16/Q6 (July 2008)