A Fast Adaptive ME Algorithm Based on H.264

Zhisheng Liu and Yuhua Peng

School of information science and engineering, Shandong University, Jinan,250100
liuzhisheng@mail.sdu.edu.cn

Abstract. This paper proposes a new fast adaptive motion estimation algorithm based on H.264. It utilizes the statistics of video sequence and the correlations of SADs in the space to early terminate the search, determines the search pattern according to the block's local motion activity, and proposes a new synthetic diamond search pattern. Compared to fast full search, the PSNR is just 0.017db lower and bitrates is 0.45% higher while the speed is 15 times up.

Keywords: fast adaptive motion estimation, block matching algorithm, stationary block.

I Introduction

The H.264/AVC standard is the latest coding standard developed by the Joint Video Team, formed by MPEG and VCEG. It is designed for a wide range of applications including low bit rate wireless video applications, SDTV&HDTV and video streaming over the Internet. The H.264/AVC standard adopts a lot of new techniques such as variable block size motion compensation, quarter-pixel accuracy motion compensation, and multiple reference frame motion compensation. In terms of compression, it provides more than 50% bit rate savings with equivalent video quality compared to the performance of MPEG4 baseline and H.263++. Because of its new techniques, the encoder's computational complexity is much increased.

Motion estimation is the primary important component in the video encoder, directly affecting the encoding speed and image quality. In H.264/AVC standard, when all of the block sizes are considered and the number of reference frame is 5 with the search window 32 using full search pattern, motion estimation consumes more than 90% of the entire encoding time. Various fast algorithms were proposed to reduce the computational complexity by reducing the num of candidate motion vectors while keeping the similar image quality to full search. The two typical methods are three-step search [1] and 2D-LOG search [2]. These algorithms limit search steps, employ rectangular search pattern and improve speed search, but the image quality is not very good. Recently, some novel algorithms were proposed such as diamond search[3],[4], cross-diamond search[5], hexagon-based search[6] and a lot of transformations of them. The UMHexagonS[7] algorithm can keep all most the same image quality with full search and is used as a reference algorithm in the JM software. Also there are many methods mainly pay attention to multireference motion estimation[8][9].

N. Sebe, Y. Liu, and Y. Zhuang (Eds.): MCAM 2007, LNCS 4577, pp. 95 – 101, 2007.

[10] proposed a FAME algorithm based on MPEG4.It utilize the technologies such as local motion activity and early terminate the search and works well on various video sequences, but it may give rise to very large motion activities.

In this article, we extend it to H.264 and propose a new algorithm based on diamond search. We use the stationary block detection and early termination to improve the speed. The LMA(local motion activity)denotes the motion activity of the current block. When the LMA is low, we use the small diamond, cross diamond and large diamond search. If the LMA is high, we propose a synthetic diamond search pattern which is better than hexagon-based search.

The rest of the paper is organized as follows. Section II describes the fast adaptive algorithm. In section III, we give the experiment results. Comparison with other algorithms is also shown in this section. We discuss about some parameters in our method in section IV. Finally, conclusion is made in section V.

2 Fast Adaptive Motion Estimation Algorithm

2.1 The Initial Search Center

There exist two selections of the initial search center. The first is just choosing the position (0, 0) as the search center in the reference frame. Although it is simple, this method is easy to be trapped in local optimization. When the global optimization is not (0, 0) and the first search step is long, it can lead to search the point that is far away from the center. Another is to predict the initial search center. There are strong correlations between adjacent frames and adjacent blocks, so many algorithms utilize this to predict the initial search center.

In this article, we choose the second one also consider the position (0, 0).

2.2 Stationary Block Detection

Experiment shows that about 98% blocks has their SAD at position (0, 0) less than 512 for MB size of 16×16[11], [12]. If the block's SAD is less than 512 at (0, 0),we can just set its MV as(0,0) and skip the motion search. Previous methods use a fixed threshold to detect stationary blocks. [10] proposed an adaptive threshold, named threshold for stationary block(TSB), which makes the detection faster and robust in the sense that it can resist the influence of noise. TSB is determined as follows:

denote by MVCL the MV candidate list which contains MVs of the upper, upper-right and left MBs;

if all adjacent MBs in MVCL have MVs at (0,0),the algorithm uses the maximum of their SADs as threshold TSB;

if one of the adjacent MBs in MVCL has MV unequal to (0,0),we use the minimum of SADs of adjacent MBs as TSB.

But when comes to H.264, it uses seven types of block motion compensation, so we to shift the TSB095. The operation is as follows:

$$TSB = \begin{cases} TSB >> 1 & \text{blocktype=2,3} \\ TSB >> 2 & \text{blocktype=4} \\ TSB >> 3 & \text{blocktype=5,6} \\ TSB >> 4 & \text{blocktype=7} \end{cases} \tag{2.1}$$

2.3 Local Motion Activity[13]

Video objects in a frame often occupy a region covered by several macroblocks, so the adjacent motion vectors are highly correlated. We compute the LMA at a macroblock position based on the adjacent macroblocks' motion vectors. The definition of LMA is as follows:

$$l_i = | x_i - \bar{x} | + | y_i - \bar{y} | \tag{2.2}$$

$$L = Max\{l_i\} \tag{2.3}$$

Where L is local motion activity measurement factor, $(\bar{x}, \bar{y})$ is the average of MVs of the upper,upper-right and left MBs, and (x_i, y_i) is their MVs.

Let LMA denotes the activity of the current MB. We categorize LMA into three classes as follows:

$$LMA = \begin{cases} Low, & \text{if } L \le 1 \\ Median & \text{if } 1 < L \le 6 \\ High & \text{if } L > 6 \end{cases} \tag{2.4}$$

If the LMA is low, it means that its adjacent MBs have the similar motion activity. The current MB and its adjacent MBs may be inside the same moving object, and may have the same MV.

2.4 Threshold for Half Stop (THS)

In the searching process, when the result is good enough, we can terminate the search. In our algorithm, we use the SADs of upper, upper-right and left MBs of the current MB as a threshold for half stop. If the LMA is low or median, we use the maximum of the SADs, else we choose the mean of them.

The reason is when the LMA is low or median, it means that the current MB and adjacent MBs are high possibly belong to one object, so the minimum SADs will be same. Otherwise, the minimum SADs will be different.

2.5 Search Pattern

There are a lot of search pattern and the diamond search pattern performs quite well among them. When the real motion vector is small, SDP, LDP and CDP are suitable. SDP is used to refine a predicted MV while EDP and LDP are used for fast wide range search in diagonal direction and in horizontal and vertical direction respectively (see Fig 1). But if the real motion vector is large, these diamond search patterns do not work well. They are easy to get trapped in local minimization. Therefore, we propose the synthetic diamond search pattern(STDP). We all know that the movement in the horizontal direction is much heavier than that in the vertical direction for natural picture sequences, the optimum motion vector can be nearly accurately predicted by an unsymmetrical synthetic diamond search in the search window, see Fig 2.

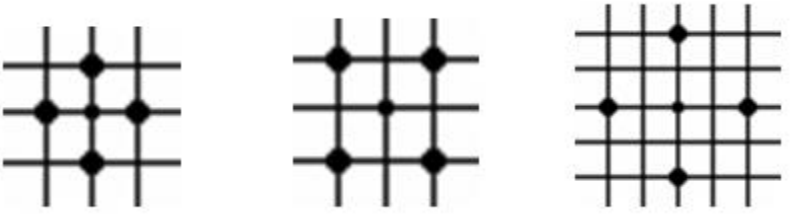

Fig. 1. SDP EDP and LDP

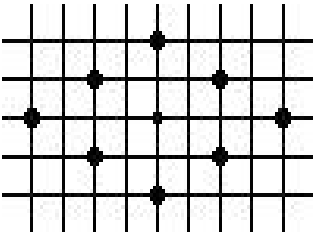

Fig. 2. STDP

2.6 Algorithm

The algorithm is described as follows:

(1). Calculate the LMA of the current MB as equations(2.2), (2.3), and get its predicted initial search point;

(2). If the LMA is low or median, the search starts from SDP, if the minimum SAD is located at the center of the diamond, the center represents the MV and the search can be terminated; else converts to EDP then to LDP. Once the minimum SAD is located at the center of the large diamond or the elastic diamond, the search goes to SDP;

(3). If the LMA is high, the search starts from STDP, if the minimum SAD is located at center, goes to SDP ; else converts to LDP.

In every step above, half stop has been used to avoid being trapped in unnecessary search.

3 Experiment Results and Comparisons

Our experiment environment is based on H.264 reference encoder of JM10, which was developed by JVT[14]. The encoder parameters are shown in table 1. All tests in the experiment are run on the Intel Pentium 4 3.6G with 512M RAM and the OS is Microsoft Windows XP. We compare our results with the FME used in JM10 and fast full search. As shown in table 2, we can see the proposed algorithm can obtain the same image quality as FME and is about 36% faster than it in various video sequences.

Table 1. Encoding Parameters

Blocktypes	QP	Num. of Reference Frame	Search Range	Encoding Pattern
7	28	5	32	IPPP

Table 2. Experiment Results

	FFS			FME			Proposed		
	PSNR (db)	BR (kbits/s)	MET (s)	PSNR (db)	BR (kbits/s)	MET (s)	PSNR (db)	BR (kbits/s)	MET (s)
news	36.77	75.35	529.790	36.72	74.97	43.619	36.72	75.30	29.543
carphone	37.29	94.00	539.710	37.28	93.65	55.772	37.27	94.37	36.594
silent	35.88	81.72	516.210	35.83	81.72	52.079	35.84	82.06	36.341
salesman	35.58	56.80	536.385	35.57	57.24	44.567	35.59	57.22	31.679
akiyo	38.19	29.36	568.642	38.21	29.50	40.104	38.19	29.46	24.803
Grandma	36.51	34.79	580.624	36.50	34.48	44.761	36.51	34.89	28.213
mobile	33.36	423.27	564.664	33.33	423.66	80.067	33.34	424.82	41.326
foreman	35.46	134.09	2463.056	35.43	133.63	300.118	35.43	135.13	188.803
stefan	35.58	1077.59	2034.869	35.56	1079.68	266.224	35.54	1088.90	181.662
tempete	34.72	1060.95	2245.258	34.70	1061.31	291.525	34.70	1062.61	175.324
sample	35.48	2077.13	2256.124	35.47	2076.66	505.208	35.48	2093.23	317.042

Table 3. The Average of Coding Efficiency

	FME	Proposed
BDPSNR(db)	-0.020	-0.017
BDBR (%)	-0.051	0.45
ME Speed Up	9.53	15.01

The average of BDPSNR, BDBR and ME speed up compared to Fast Full Search (FFS) are shown in table 3. With our algorithm, the PSNR is just 0.017db lower and bitrates is 0.45% high while the speed is 15 times up.

4 Discussion

In section II, we refer to TSB (threshold for stationary block) and THS (threshold for half stop). The effects of them were already discussed in [10]. An increase in the

upper bound of TSB and THS leads to small number of check points involved. However, the PSNR decreases while the Bit Rate increases. There is a tradeoff between video quality and encoding time. Now we give the experiment results in H.264, see table 4.

Table 4. The Effect of Upper-bounds of TSB and THS

TSB and THS's value	512			768			1024		
	PSNR	BR	MET	PSNR	BR	MET	PSNR	BR	MET
carphone	37.27	94.37	36.594	37.22	94.78	28.522	37.09	97.08	22.933
news	36.72	75.30	29.543	36.71	75.36	19.621	36.68	75.78	15.904
foreman	35.43	135.13	188.803	35.40	135.60	172.103	35.35	136.09	150.209
silent	35.84	82.06	36.341	35.84	82.44	32.395	35.83	83.19	23.569
salesman	35.59	57.22	31.679	35.59	57.29	26.155	35.57	57.30	18.447
sample	35.48	2093.23	317.042	35.47	2093.00	313.531	35.47	2098.43	285.852
stefan	35.54	1088.90	181.662	35.54	1090.59	157.572	35.53	1094.78	150.602
akiyo	38.19	29.46	24.803	38.18	29.45	15.850	38.11	30.46	12.655
grandma	36.51	34.89	28.213	36.50	34.66	23.377	36.48	34.73	16.407
tempete	34.70	1062.61	175.324	34.69	1063.40	152.746	34.66	1066.20	137.850
mobile	33.34	424.82	41.326	33.35	424.55	39.768	33.33	425.51	37.364

5 Conclusions

In video encoding software, the motion estimation is very important because it consumes most of the encoding time. So we propose a fast adaptive motion estimation algorithm. From the experiment results we know that it can save 36% ME time compared with FME while keeping the same image quality and similar coding efficiency. Our algorithm performs well both on large dynamic motion variation sequence and simple uniform motion video.

Reference

1. Koga, T., Ilinuma, K., Hirano, A., Iijima, Y., Ishiguro, T.: Motion compensated interframe coding for video conference. In: Proc. Nat. Telecommun. Conf., November 1981, New Orleans, LA, (1981)
2. Jain, J.R., Jain, A.K.: Displacement measurement and its application in interface image coding, IEEE Trans. Commun., COM-29(12) (December 1981)
3. Zhu, S., Ma, K.K.: A new diamond search algorithm for fast block-matching motion estimation. In: Proc, Int. Conf. Information, Communication and Signal Processing, September 9-12, 1997, vol. 1, pp. 292–296 (1997)
4. Tham, J.Y., Ranganath, S., Ranganath, M., Kassim, A.A.: A novel unrestricted center-biased diamond search algorithm for block motion estimation. IEEE Trans. Circuits Syst. Video Technol (August 1998)
5. Cheung, C.H., Po, L.M.: A novel cross-diamond search algorithm for fast block motion estimation. IEEE Trans. Circuits Syst. Video Technol, 12 (December 2000)

6. Zhu, C., Lin, X., Chau, L.P.: Hexagon-based search pattern for fast block motion estimation. IEEE Trans. Circuits Syst. Video Technol, 12 (May 2002)
7. Chen, Z., Xu, J.F., Zhou, P., He, Y.: Hybrid unsymmetrical-cross multi-hexagon-grid search strategy for integer pel motion estimation in H.264. In: Proceedings of PCS, San Malo (2003)
8. Kim, S.E., Han, J.K., Kim, J.G.: An efficient scheme for motion estimation using multi-reference frames in H.264/AVC. In: IEEE Trans. Multimedia, vol. 8(3), (June 2006)
9. Huang, Y.W., Hsieh, B.Y., Chien, S.Y., Ma, S.Y., Chen, L.G.: Analysis and complexity reduction of multiple reference frames motion estimation in H.264/AVC. IEEE Trans. Circuits Syst. Video Technol, 16(4) (April 2006)
10. Ahmad, I., Zheng, W., Luo, J., Liou, M.: A fast adaptive motion estimation algorithm. IEEE Trans. Circuits and systems for video tech. 16(3) (March 2006)
11. Hosur, P.I., Ma, K.K.: Motion vector field adaptive fast motion estimation. the Second Int. Conf. Inf, Commun., Signal Process., Singapore (December 1999)
12. Optimization Model Version 1.0, ISO/IEC JTC1/SC29/WG11 N3324 (March 2000)
13. Hosur, P.I.: Motion adaptive search for fast motion estimation. IEEE Trans. Consumer Electronics, 49(4) (November 2003)
14. JM10,http://iphome.hhi.de/suehring/tml/download/old_jm/jm10.zip
15. Wiegand, T.: JVT-K051, Joint Video Team (JVT) of ISO/IEC MPEG &ITU TVCEG ISO/IEC JTC1/SC29/WG11 and ITU T SG16 Q.6 , In: 12th Meeting, 17-23 July 2004: Redmond, WA, USA (2004)