Journal of Zhejiang University SCIENCE A
ISSN 1009-3095 (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

A fast motion estimation algorithm for mobile communications

CHEN Guo-bin

(*Department of Information and Electronic Engineering, Zhejiang University, Hangzhou 310027, China*)

E-mail: chenguobin@zju.edu.cn

Received Dec. 1, 2005; revision accepted Feb. 8, 2006

Abstract: The limitation of processing power, battery life and memory capacity of portable terminals requires reducing encoding complexity in mobile communications. Motion estimation (ME) is the most computationally intensive module in a typical video codec, which determines not only the encoder's performance but also the reconstructed video quality. In this paper, a fast ME algorithm for H.264/AVC baseline profile coding is proposed based on the analysis of motion vector field and error surface, and the statistical distributions of different type macroblocks (MBs). Simulation results showed that: in comparison with MVFAST, the proposed algorithm can decrease the computational load over 7.2% with no requirement of expanding memory capacity while maintaining the same video quality as MVFAST. Furthermore, its simplicity makes it easy to be implemented on hardware.

Key words: Motion estimation (ME), Error surface, Motion vector (MV)
doi:10.1631/jzus.2006.AS0013 **Document code:** A **CLC number:** TN919.8

INTRODUCTION

With the development of mobile communications and multimedia techniques in recent years, the traditional audio service no longer satisfies the ongoing demand for better communications, and a new service called MMS (multimedia messaging service) incorporating audio, video and text appears. In this multimedia communication system, video communication deals with higher data amount and higher computational load than other media do. These characters pose great challenges for the time-critical mobile video communications. For example, an MPEG-4 simple profile codec has been implemented on a Texas Instruments' TM3205410 40 MHz processor (Budagavi *et al*., 2000). Profiling results showed that this codec cannot achieve real-time performance, even when using SQCIF sequences, it can encode only 1 frame/s, and can decode only about 20 fps (Al-Mualla, 2002). Another example is the implementation of H.264 baseline profile on more powerful DM642 600 MHz. Again this implementation cannot achieve real-time processing, as it can only encode CIF images about 4 fps.

H.264/AVC is the newest international standard (ISO/IEC, 2003), which represents the state-of-the-art in current video coding. It is reported that H.264 can achieve 50% coding gain over previous standards MPEG-2 and H.263 (Kamaci and Altunbasak, 2003). Many developers choose this standard as their encoder core. Just as previous video coding standards, it employs motion compensation/transform mode to eliminate spatio-temporal redundancy. It is not difficult to show that the high computational complexity of the codec is due mainly to the motion estimation process. It consumes over 80% of total encoding time. Thus, by reducing the complexity of this process, the overall complexity of the codec can be reduced.

Many fast motion estimation algorithms have been proposed, such as three-step search, new four-step search and diamond search (Li *et al*., 1994; Po and Ma, 1996; Furht *et al*., 1997). All these algorithms can reduce complexity efficiently, although they do not consider the continuity of the motion vector field. If there exists complex motion in encoding sequences, image quality drops dramatically. Some high performance algorithms such as MVFAST, PMVAST (ISO/IEC, 2000; Zhu and Ma, 2000) can make a good compromise between computation complexity and video quality. They take not only the continuity of motion vector field into account, but also the motion activity degree. By using two diamond modes, they can speed up the search process while maintaining almost the same image quality as full search. Whereas, based on in-depth analysis of MVFAST and PMVFAST search process, we find there are a lot of repetitive searches. To eliminate these repetitions, more memory space is required to store the search record. In this paper, a low complexity motion estimation algorithm to eliminate the repetitive search is proposed without the requirement of expanding memory capacity. A new feature of H.264 is that it employs variable block size motion compensation, which greatly increases the computational complexity. The computation is approximately in proportion to the search mode. Full search of all modes may not be available for time critical application, especially for mobile communications. In consideration of the constraints of mobile communications, only fewer primary modes are present on the basis of the statistical information collected under low bitrate, which greatly decreases the total computation.

This paper is organized as follows. Section 2 provides some technical preliminaries of ME in H.264. The proposed algorithm is set forth in Section 3. Finally, in Section 4, practical results are given and analyzed, and conclusions are drawn in Section 5.

PRINCIPLES OF MOTION ESTIMATION

The most commonly used ME method is the block-matching motion estimation (BMME), adopted by mainstream video coding due to its simplicity and easy implementation. In this algorithm, the current frame is first divided into blocks. The motion of each block is then estimated by searching for the best-match block in the reference frame according to some distortion measure, such as sum of absolute difference (SAD) or sum of square difference (SSD). SAD is specified as follows:

$$SAD_{M\times N}(v_x,v_y)=\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\left|f_k'(i+v_x,j+v_y)-f(i,j)\right|, \quad (1)$$

where M and N are the width and height of block respectively; $f(i,j)$ is the block to be coded; $f_k'(i,j)$ is the candidate block in the reference frame k. This search is usually restricted to a search window around the corresponding block in the reference frame. The displacement (v_x,v_y) between the block and best-match block is called motion vector (MV). The decoder uses these MVs and corresponding error residuals to reconstruct the video. A typical ME diagram is illustrated in Fig.1.

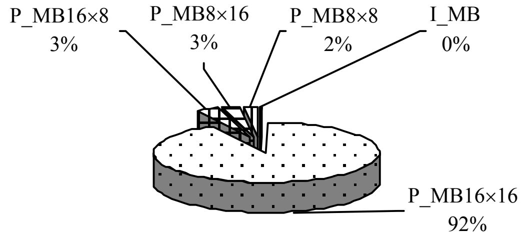

Fig.1 Distributions of MB coding types in P-slice

To achieve the best encoding performance, Lagrangian operator is often used to find the optimal motion vectors (v_x,v_y) for a block size of $M\times N$. The cost function is defined as follows:

$$J_{M\times N}(v_x,v_y)=SAD_{M\times N}(v_x,v_y)+\lambda R(MVD_{M\times N}(v_x,v_y)), \quad (2)$$

where $MVD_{M\times N}(v_x,v_y)$ is derived as follows:

$$MVD_{M\times N}(v_x,v_y)=(v_x-\hat{v}_x,v_y-\hat{v}_y), \quad (3)$$

where $(\hat{v}_x,\hat{v}_y)$ is the prediction MV.

PROPOSED ALGORITHM

In H.264 baseline profile, only I and P slice types may be present in the syntax. I slice is used for refreshing after a long distance coding and consumes much less processing cycles than P slice does. For each MB in P slice, there exist many coding types as it can be partitioned into variable block size. Full selection of all the coding types provides the best coding results but with the most computational complexity. In consideration of the low computation power of mobile terminals, a simplification must be done on it. After extensive experiments under low bitrate, we found the most primary coding type is MB 16×16, it consumes about 90% processing cycles. Fig.1 illustrates the distributions of each MB coding types in the slice. In our scheme, we only use the most three primary partitions, i.e., 16×16, 16×8 and 8×16. The impact of this scheme can be referred to in the experiments results in Section 4.

Many algorithms, such as three-step, new three-step, and spiral search, assume that the error surface is usually unimodal and that the distribution of the motion vector is center-biased. Typical error surface and prediction center-based motion vector distribution are illustrated in Fig.2 and Fig.3 respectively. As we see, these assumptions are always true if we can predict the search center accurately. Our algorithm adopts some ideas from MVFAST and the scheme in (Ting *et al*., 2003) and employs only small diamond search pattern. To get an early termination of search, a threshold T is set according to its size. For a block with size of $M\times N$, its early termination threshold is equal to 2 multiplied by its pixel number, i.e., $2\times M\times N$.

In-depth looking at MVFAST, we found that if previous two motion directions are determined, repetitive search can be avoided. Except for the first diamond, consequent searches only need 2 or 3 points, not 4 points in MVFAST; if previous two moving direction are both towards left, current search need search 3 points; if previous two directions are not identical, only 2 points are searched. Fig.4 displays a typical search process.

The first step of our strategy is to determine the search mode and search center. If the coding types of above and left MB are different, all modes are used. If the coding types of the above and left MB are equal, the same types are used in searching. We use the prediction MV, the MVs of the left MB, the top MB, the topright MB, the co-located MB in reference frame as the search center candidate for MB 16×16; In addition to aforementioned candidates, the best MV of MB 16×16 is added to candidate set for MB 16×8 and MB 8×16. The vector yielding minimum

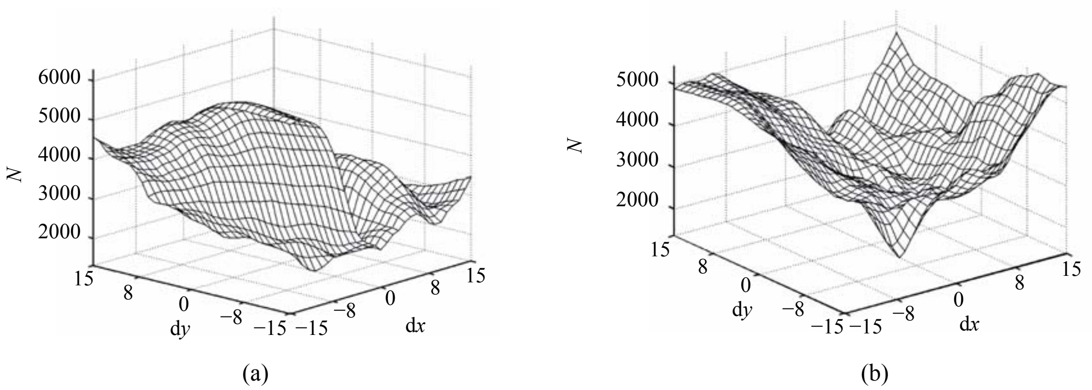

Fig.2 Error surface. (a) Foreman @10 fps; (b) Coastguard @10 fps

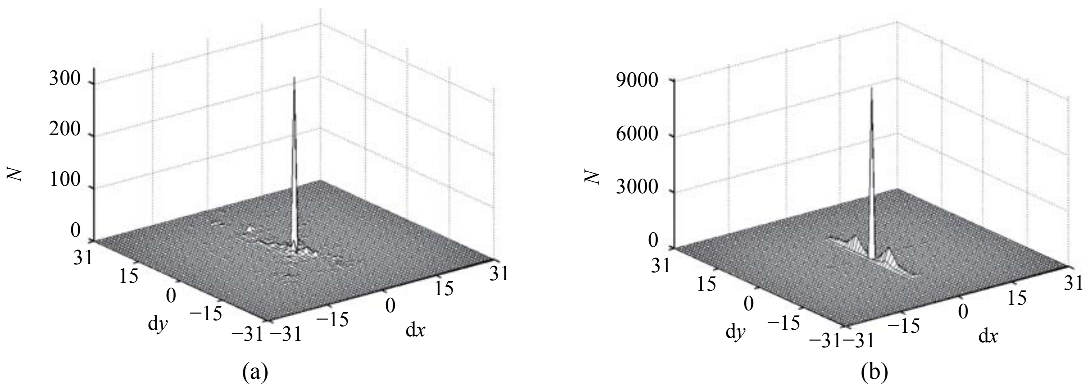

Fig.3 Prediction center-biased distribution of MV. (a) Foreman @10 fps; (b) Coastguard @10 fps

$J_{M\times N}(v_x,v_y)$ is chosen as the search center.

Next, we used repetition-eliminated search to find the optimal MV. Diamond search pattern is centered at the search center, and all the checking points of diamond are tested. If the center position yields the minimum $J_{M\times N}(v_x,v_y)$, then the center represents the motion vector. Otherwise do Step 2 recursively untill the diamond is beyond the search region.

SIMULATION RESULTS

To evaluate the performance of the proposed algorithm, we integrate it into the reference software T264.0.14. Some important parameters are set as follows: (1) Sequence type is IPPP…; (2) Search range is 16×16; (3) Entropy coding method is CAVLC; (4) No rate control, the QP of all slice is set to 30. Four benchmark sequences, Carphone, Coastguard, Foreman and Stefan, are tested in our simulations. Each sequences has 100 frames; Carphone and Foreman are at the format of QCIF @30 fps; and Coastguard and Setfan at the format of CIF @30 fps; We compare the full search (FS) and MVFAST algorithm with the proposed algorithm.

Fig.5 illustrates the performance comparison between different algorithms. We can see that our proposed algorithm can be maintained at almost the

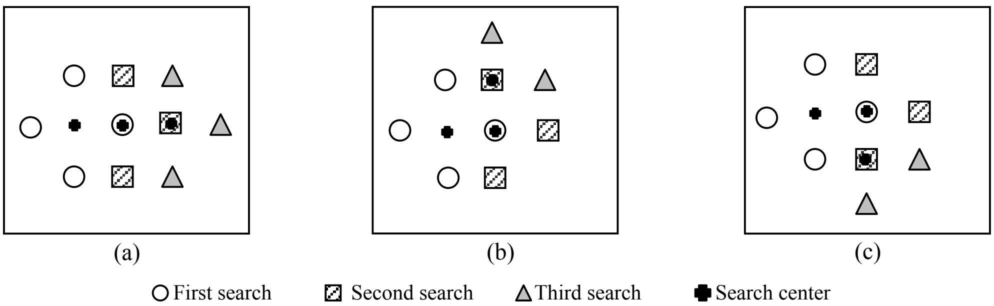

Fig.4 Search patterns and process. (a) right & right; (b) right & up; (c) right & down

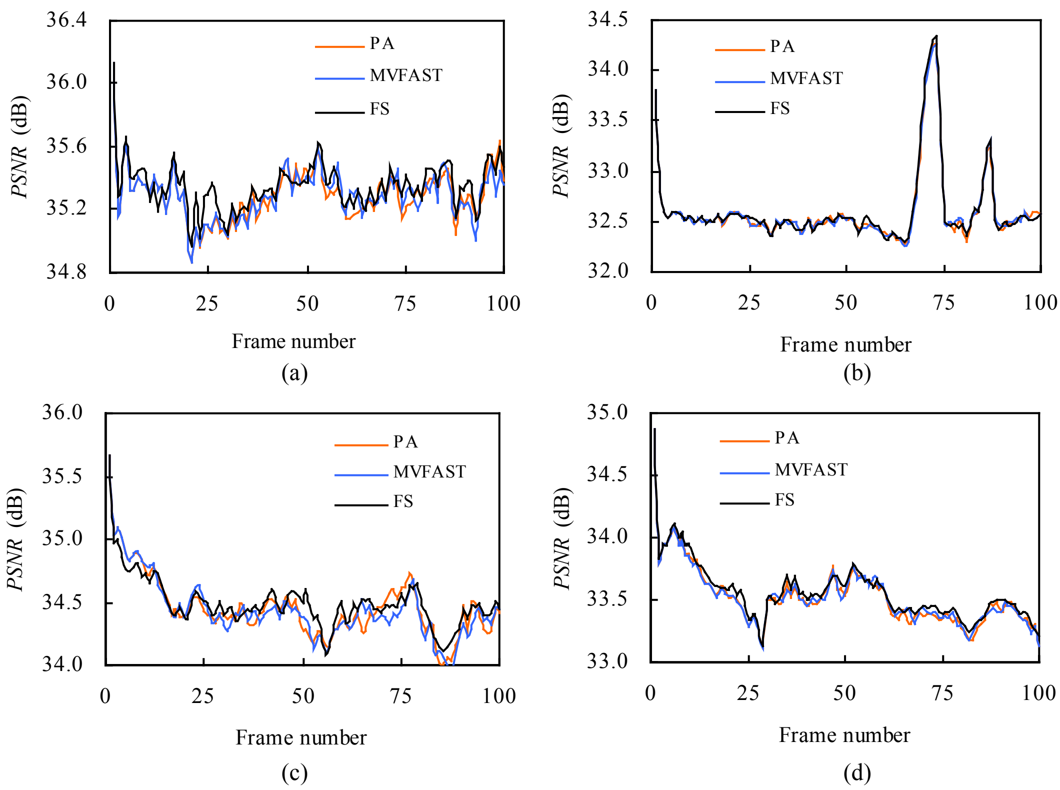

Fig.5 *PSNR* comparison. (a) Carphone-QCIF; (b) Coastguard; (c) Foreman-QCIF; (d) Stefan

same quality as FS and MVFAST. The maximum deviation in all sequences between FS and PA is only about 0.2 dB. Table 1 displays a detailed performance comparison of PSNR, Speedup and bitrate increase. For clarity, we compared the checking points of each coding type. Our proposed algorithm has a higher speedup than MVFAST. If we define one MB 16×16 search complexity as 1, and MB 16×8 and MB 8×16 complexity as 0.5, the final complexity of MVFAST on Carphone, Coastguard, Foreman and Setfan are 4110.75, 22368.8, 4678.4 and 22607.85 respectively; and that of PA is 3814.9, 1978.28, 4250.75 and 19893.15. Consequently, we get the speedup gains of 7.2%, 11.2%, 9.14 and 12.0%. Compared with FS, PA only causes bitrate increase of –1.1%, 0.01%, 0.75% and 1.47%. The bitrate increases slightly.

CONCLUSION

This paper presents a low complexity motion estimation algorithm for mobile communications on H.264/AVC baseline profile. The MB statistical distribution of each MB type is colleted on the basis of extensive experiments. It was found that over 90% MBs are coded in size 16×16 predicted MB. Considering the low processing power of mobile terminals, we only use the 3 primary MB sizes in our encoder. In the motion estimation process, a modified MVFAST algorithm is used which can eliminate repetitive searches without requirement of enlarging memory space. Simulation results showed that compared with MVFAST, the proposed method can not only get a faster encoding speed but also yield a better image

Table 1 Performance comparison between different algorithms

Sequence	Item	Motion estimation algorithm								
		FS			MVFAST			PA		
		16×16	16×8	8×16	16×16	16×8	8×16	16×16	16×8	8×16
Coastguard Carphone-QCIF	Check points	323433	215622	215622	2608.5	1496.3	1508.2	2327.0	1486.5	1489.3
	Speedup	1	1	1	124.0	144.1	142.0	138.9	145.1	144.8
	PSNR		35.3516			35.2983			35.2979	
	Δ*PSNR*		0			–0.0533			–0.0537	
	Bitrate (kbps)		82.08			81.499			81.204	
	Δbitrate (%)		0			–0.7			–1.1	
Coastguard	Check points	1293732	862488	862488	15234.4	7113.475	7155.3	12881.3	6884.6	6918.4
	Speedup	1	1	1	84.9	121.2	120.5	100.4	125.3	124.7
	PSNR		32.6123			32.6042			32.6064	
	Δ*PSNR*		0			–0.0081			–0.0059	
	Bitrate (kbps)		979.658			979.442			980.632	
	Δbitrate (%)		0			–0.02			0.01	
Foreman-QCIF	Check points	323433	215622	215622	3009.6	1673.1	1664.5	2613.7	1639.2	1634.5
	Speedup	1	1	1	107.5	128.9	129.5	123.7	131.5	131.9
	PSNR		34.4870			34.4391			34.4542	
	Δ*PSNR*		0			–0.0479			–0.0328	
	Bitrate (kbps)		116.954			117.880			117.835	
	Δbitrate (%)		0			0.79			0.75	
Stefan	Check points	1293732	862488	862488	15686.2	6924.7	6918.6	13106.2	6790.4	6783.5
	Speedup	1	1	1	82.5	124.6	124.7	98.7	127.0	127.1
	PSNR		33.5612			33.5243			33.5217	
	Δ*PSNR*		0			–0.0369			–0.0395	
	Bitrate (kbps)		914.128			927.055			927.547	
	Δbitrate (%)		0			1.41			1.47	

quality, while the computational complexity drops over 7.2%. Furthermore, pixel decimation can be incorporated into this algorithm to get more speedup. Its simplicity and low complexity make it easy to implement on hardware and suitable to be used in portable terminals.

References

Al-Mualla, M.E., 2002. Video Coding for Mobile Communications: Efficiency, Complexity, and Resilience. Academic Press.

Budagavi, M., Heinzelman, W.J., Talluri, R., 2000. Wireless MPEG-4 video communication on DSP chips. *IEEE Signal Processing Magazine*, **17**(1):36-53. [doi:10.1109/79.814645]

Furht, B., Greenberg, J., Westwater, R., 1997. Motion Estimation Algorithms for Video Compression. Kluwer Academic Publishers.

ISO/IEC, 2000. N3324, Optimization Model.

ISO/IEC, 2003. Information Technology—Coding of Audio-Visual Objects, Part 10: Advanced Video Coding. Switzerland.

Kamaci, N., Altunbasak, Y., 2003. Performance Comparison of the Emerging H.264 Video Coding Standard with the Existing Standards. Proceeding of IEEE International Conference on Multimedia and Expo (ICME 2003), p.345-348.

Li, R., Zeng, B., Liou, M.L., 1994. A new three step search algorithm for block motion estimation. *IEEE Trans. on Circuits and Systems for Video Technology*, **4**(4):438-442. [doi:10.1109/76.313138]

Po, L.M., Ma, W.C., 1996. A novel four-step search algorithm for fast block motion estimation. *IEEE Trans. on Circuits and Systems for Video Technology*, **6**(3):313-317. [doi:10.1109/76.499840]

Ting, C.W., Po, L.M., Cheung, C.H., 2003. Center-Biased Frame Selection Algorithms for Fast Muliti-Frame Motion Estimation in H.264. Proceeding of IEEE International Conference on Neural Network & Signal Processing. Nanjing, China, p.1258-1261.

Zhu, S., Ma, K.K., 2000. A new diamond search algorithm for fast block matching motion estimation. *IEEE Trans. on Image Processing*, **2**:287-290.