An H.264 Spatio-temporal Hierarchical Fast Motion Estimation Algorithm for High-Definition Video

Yu-Shin Cheng[1], Zong-Yi Chen[2], and Pao-Chi Chang[2]
[1]Telecommunication Laboratories, Chunghwa Telecom Co., Ltd., Yang-Mei, Taoyuan 32601, Taiwan, R.O.C.
[2]Department of Communication Engineering, National Central University, Jhongli, Taoyuan 32001, Taiwan, R.O.C.
[1]yscheng@cht.com.tw, [2]{zychen, pcchang}@vaplab.ee.ncu.edu.tw

***Abstract*—In the advanced H.264 video coding, the computation complexity is much higher than the previous video coding standards due to the variable block size and multi-reference frame features which are used in the motion compensation process. This paper proposes a hierarchical H.264 fast motion estimation algorithm to decrease the coding complexity in both spatial and temporal domains for encoding high-definition videos. In the spatial domain, we utilize the fast search method with a hierarchical-subsampling structure to decrease the memory access bandwidth of search points. In the temporal domain, we employ the linear motion model to further reduce the search ranges of multiple reference frames. This search algorithm is particularly suitable for being implemented in the parallel-processing architecture with the limited hardware resources. Simulation results show that the proposed algorithm can reduce up to 98.2% computation complexity of Full Search in JM with less than 0.1 dB video quality degradation.**

I. INTRODUCTION

H.264/AVC [1] provides very high coding efficiency compared to previous video coding standards. With the variable block size and multiple reference frames in the motion compensation process, H.264/AVC could remove most of the visual redundancy by inter prediction. However, the computation complexity will be increased significantly with the growth in number of coding modes and reference frames if the full search method is applied. By experiments in the JM12.4 reference software [2], we can find that the motion estimation (ME) process usually occupies over 80% of total computation complexity. The situation is even more serious while compressing the high-definition (HD) video sequences. Thus, plenty of researches focus on the fast ME algorithms to reduce the computation burden [3][4].

Among existing fast ME algorithms, many of them are designed without the consideration of resource limitations. Thus, they are impractical to be implemented directly on the hardware architecture. The main considerations in the hardware design process are the area cost, the cycle count, and the bandwidth. To decrease the area cost, the internal memory as well as the number of processing units (PUs) should be reduced. The cycle count could be handled by the design of parallel processing and pipelining. As for the bandwidth consideration, the access bandwidth could be decreased effectively by subsampling because of the high correlation with the search range in the ME process.

The literature shows that the Parallel Multi-Resolution ME (PMRME) algorithm [5] provides a practical hardware solution to HD video coding. With a three-layer structure, the access frequency between the internal and external memory will be drastically reduced. Besides, the three layers can be processed independently. The first layer (Layer 0) does not go through any subsampling process. Full search is applied with the search center at the median predictor. Because most of the best motion vectors (MVs) are distributed around the median predictor, the PMRME algorithm spends most of the computation complexity on Layer 0. As for the second (Layer 1) and third layer (Layer 2), the best MV that is farther away from the median predictor occurs less often, so on average the PMRME algorithm spends less computation complexity on these two layers. The full search is applied to both layers to find the best MV.

For *station2*, Table I shows the statistical result of the best MV distribution on different layers in PMRME algorithm. Similar results can be found in other sequences. We can observe that over 90% of the best MVs are contributed by Layer 0. That means less than 10% of the best MVs are contributed by Layer 1 and Layer 2 together. However, the full search is still applied on these two layers in the original design. In addition, according to the statement in [6], over 85% of the best MVs are contributed just by the previous one reference frame. Thus, this paper proposes a spatio-temporal hierarchical fast motion estimation algorithm (STHME) to further reduce the computation complexity in the ME process for HD video coding.

In the following, Section II will describe the proposed STHME algorithm in detail. The comparisons of JM12.4, PMRME, and STHME simulation results are shown in Section III. Conclusions are given in the final section.

II. PROPOSED SPATIO-TEMPORAL HIERARCHICAL FAST MOTION ESTIMATION ALGORITHM

A. Fast ME design in spatial domain

In PMRME algorithm, the total contributions of the best MVs from Layer 1 and Layer 2 are less than 10%. Therefore, in our algorithm, we merge Layer 1 and Layer 2 into a single layer, denoted as L_1, and further adopt the fast search algorithm suitable for encoding the HD video. Because of the requirement of larger search range of HD video coding, most

978-1-4244-3828-0/09/$25.00 ©2009 IEEE

TABLE I. BEST MV DISTRIBUTION OF PMRME ALGORITHM FOR *STATION2*

QP	Layer 0	Layer 1	Layer 2
12	83.02%	14.77%	2.20%
16	89.85%	8.89%	1.26%
20	94.91%	4.31%	0.77%
24	96.10%	3.19%	0.70%
28	96.46%	2.84%	0.70%
32	96.71%	2.59%	0.71%
36	96.46%	2.69%	0.85%
40	95.81%	2.96%	1.23%
44	95.15%	3.27%	1.58%
Average	**93.83%**	**5.06%**	**1.11%**

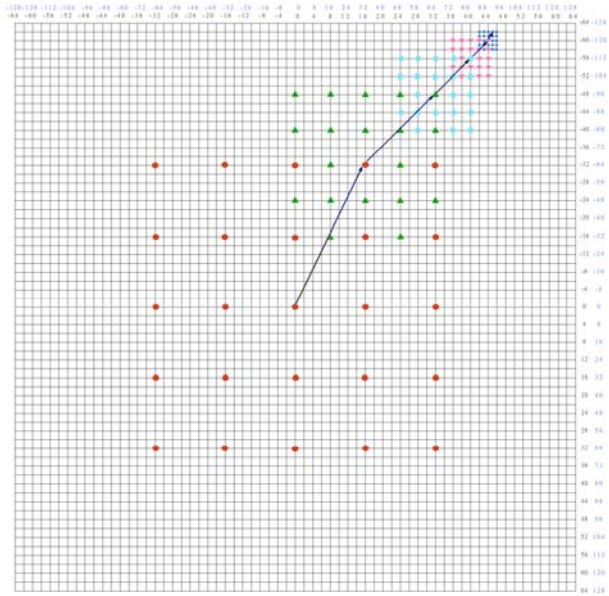


Figure 1. An example of 25-point search pattern.

of the traditional fast search algorithms, such as four-step search (4SS) [7], are not suitable to be applied directly. According to the description in [8], there are two ways to refine the conventional fast search algorithms to fit the much larger search range of HD videos. One way is to increase the number of search steps, and the other is to enlarge the search pattern. In our proposed algorithm, we choose the latter one which is more suitable from the hardware viewpoint because of the less search steps. In this work, the number of search candidates for each search step is 25, and the number of search steps for each layer depends on the search range. Fig. 1 shows an example of 25-point search pattern performed in ±64 search range. The distance between every two search candidates decreases step by step.

In STHME algorithm, the search center of the first layer, denoted as L_0, is determined by the median predictor, and the search range is ±8. All inter modes are examined with the full search. The search process on L_0 performs in original resolution without subsampling. As for L_1, the resolution is subsampled from L_0 by the factor of two both horizontally and vertically. To retain the HD (1920×1080) video quality,

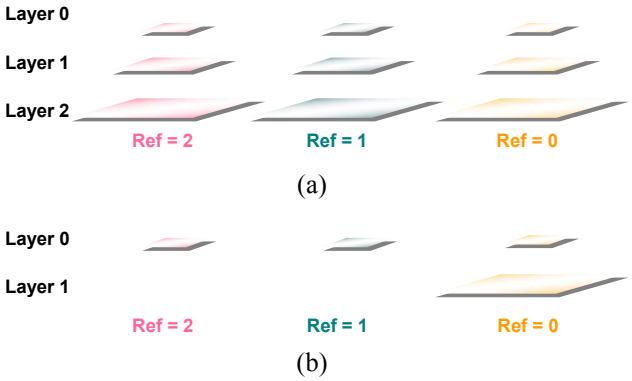


Figure 2. The spatio-temporal structure: (a) PMRME; (b) STHME.

the lower bound of search range is typically ±128. Hence the search range after subsampling is set to ±64 on L_1 here. Only the inter modes 1~4 have to be tested, and the fast search algorithm with 25-point search pattern is applied. Thus, the best integer MV can be determined within five search steps. To avoid the search result being trapped into a local minimum, we employ two search centers on L_1. One is the median predictor, and the other is the origin.

B. Fast ME design in temporal domain

Fig. 2(a) shows the spatio-temporal structure of PMRME algorithm. All the reference frames use the same 3-layer hierarchical structure in the ME process. Because over 85% of the best MVs are distributed on the just previous one reference frame, we propose a different search structure on multiple reference frames to further reduce the computation complexity. Fig. 2(b) shows the spatio-temporal structure of STHME algorithm. We not only simplify the 3-layer structure of PMRME into 2-layer in the spatial domain, but also try to accelerate in the temporal domain. If the reference frame is the previous one, the full search with ±8 search range is performed on L_0 and the 5-step fast search with the 25-point search pattern is performed on L_1. If the reference frame is not the previous one, we keep the full search on L_0 only and skip the search on L_1. However, it is usually not sufficiently accurate to use the original median predictor for all reference frames because the best matched block may be outside the ±8 search window. Therefore, we have to develop a new approach to obtain a better initial search center, called multi-frame linear motion predictor, to increase the accuracy of predictor on L_0.

Generally, most of the objects tend to move linearly in consecutive video frames, so as for the corresponding MVs. Consequently, we apply the linear motion assumption to the multi-frame motion predictors in STHME algorithm. First, we define MV_{real} as the best MV determined by the ME process on the K^{th} reference frame and MV_{norm} as the MV after proceeding by the normalization. The normalization is processed as follows: (1) if the current block is an intra block, $MV_{norm} = (0,0)$; (2) if the current block is an inter block, $MV_{norm} = \frac{1}{K} MV_{real}$. Second, we define MVP_C^i as the MV predictor after the median process expressed by (1).

$$MVP_C^{\,i} = median(MV_L^{\,i}, MV_R^{\,i}, MV_U^{\,i}, MV_D^{\,i}, MV_C^{\,i}) \quad , i = 0 \sim 15 \tag{1}$$

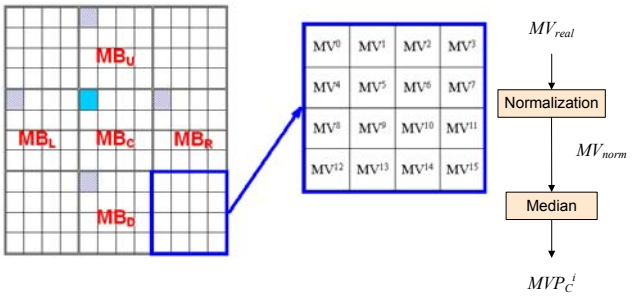
The parameter i in (1) denotes the i^{th} 4×4 block MV in a macroblock (MB). Refer to Fig. 3(a), $MV_L^{\,i}$, $MV_R^{\,i}$, $MV_U^{\,i}$, $MV_D^{\,i}$, and $MV_C^{\,i}$ belong to MB_L, MB_R, MB_U, MB_D, and MB_C, respectively. MB_L, MB_R, MB_U, and MB_D are the MBs at left side, right side, upper, and lower positions corresponding to the current MB MB_C. Please note that since the MV information is stored per 4×4 block in H.264, there are 16 MVs in each MB. Fig. 3(b) illustrates the whole process mentioned above.

After obtaining MVP_C^i from (1), we can define the multi-frame linear motion predictors as $2\times MVP_C^i$ and $3\times MVP_C^i$ for Ref=1 and Ref=2, respectively. Fig. 4 shows the temporal relations between MVP_C^i and the multi-frame linear motion predictors. At $T=t-2$, $MVP_C^i(N-2)$ can be obtained by (1). At $T=t-1$, the initial search center is $2\times MVP_C^i(N-2)$ while performing L_0 search on the second previous reference frame, i.e., *N-3*, and the search center is $3\times MVP_C^i(N-2)$ while performing L_0 search on the third previous reference frame, i.e., *N-4*. After the whole ME process at $T=t-1$, the $MVP_C^i(N-1)$ is also obtained. Table II shows the absolute difference of distance between the proposed multi-frame linear motion predictor and best MV determined by the original JM for the second and third previous reference frames, denoted by *MV1_diff* and *MV2_diff*, respectively. The test sequence is *pedestrian_area*. For instance, the case that the distance is within 1 pixel between the proposed predictor and best MV obtained from the second previous reference frame by JM counts to 64.48% (31.52 + 32.96). We can find that more than 80% MV predictors obtained by the proposed method can hit the original best MVs within ±8 pixels.

Table III summarizes the detailed settings of STHME algorithm. If the reference frame is not the previous one, we always use the proposed multi-frame linear motion predictor as the initial search center in L_0. Moreover, L_1 searches are also skipped. Most importantly, by employing the multi-frame linear motion predictor, the multi-frame search can be performed simultaneously and independently, not like the conventional frame-by-frame approach [6]. Note that all layers in STHME can be processed in parallel and especially suitable for hardware implementation. In order to obtain the multi-frame linear motion predictor, the overhead we should afford is only to additionally record the MV information of the previous one frame.

III. Experimental Results

The simulations are performed with JM12.4 baseline profile. The sequence type is IPPP, and the frame rate is 25 fps. The number of reference frames used in P prediction is 3 and QPs are 12, 16, 20, 24, 28, 32, 36, 40, and 44. Low complexity RDO mode is used. Total 100 frames are coded for each sequence. The test sequences include *station2*, *pedestrian_area*, and *rush_hour* in the size of 1920×1072.



(a) (b)

Figure 3. Motion vector prediction in STHME: (a) MB and MV notations; (b) The calculation process of MVP_C^i.

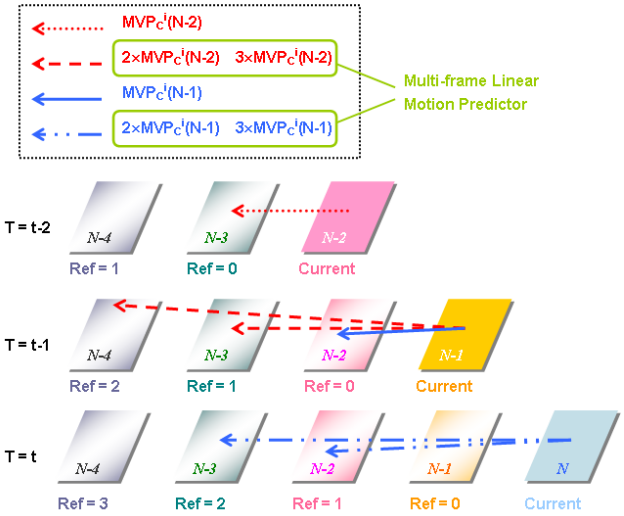


Figure 4. The multi-frame linear motion predictor.

TABLE II. ABSOLUTE VALUE OF *MV1_diff* AND *MV2_diff*

pedestrian_area					
Distance	**0**	**0.0~1.0**	**1.0~2.0**	**2.0~3.0**	**3.0~4.0**
MV1_diff	31.52%	32.96%	9.84%	3.47%	2.24%
MV2_diff	33.14%	26.83%	10.60%	4.82%	2.71%
Distance	**4.0~5.0**	**5.0~6.0**	**6.0~7.0**	**7.0~8.0**	**8.0+**
MV1_diff	1.37%	0.70%	0.57%	0.71%	16.62%
MV2_diff	1.64%	0.96%	0.65%	0.46%	18.17%

TABLE III. SUMMARY OF STHME ALGORITHM

Ref. Frame	**0**		**1**	**2**
Layer	0	1	0	0
Initial Search Center	Median Predictor	(0,0) and Median Predictor	Multiframe Linear Motion Predictor	Multiframe Linear Motion Predictor
Range	±8	±128	±8	±8
Mode	1~7	1~4	1~7	1~7
Search Method	Full Search	5-Step Search	Full Search	Full Search

All experiments are executed on a PC with Intel Core 2 Extreme QX9650 3.67 GHz CPU, 8G DDR II 800MHz RAM, and the OS is Windows XP x64 version.

Fig. 5 shows the R-D curves of *station2* and *rush_hour*. If QP<36, the R-D performances of PMRME and STHME are very similar to that of JM12.4. The performances of PMRME and STHME are better than JM12.4 with QP=12 due to the rough approximation of Lagrange Multiplier in JM12.4 codec while the low complexity RDO is used. The performances of PMRME and STHME are worse than JM12.4 with QP>36 due to the serious blocking effect.

Table IV shows the encoding time and R-D performance results of PMRME and STHME compared to the full search of JM12.4. The assessment criteria are defined by (2), (3), and (4). Although PMRME algorithm shows a little bit better R-D performance than the proposed STHME in average, the run time is longer. STHME algorithm takes the shortest run time with the negligible quality loss and bit-rate increase. The run time of STHME algorithm is 43% of PMRME and 1.8% of JM12.4 only. Please note that the time shown in Table IV is calculated by accumulating all layers in serial processing. The coding time could be decreased significantly if STHME is implemented in parallel by hardware.

$$\Delta PSNR = PSNR_{new} - PSNR_{JM} \tag{2}$$

$$\Delta bit\_rate = \frac{bit\_rate_{new} - bit\_rate_{JM}}{bit\_rate_{JM}} \times 100\% \tag{3}$$

$$run\_time = \frac{run\_time_{new}}{run\_time_{JM}} \times 100\% \tag{4}$$

IV. Conclusions

This paper proposes a hierarchical H.264 fast motion estimation algorithm for the HD video by decreasing the coding complexity in both spatial and temporal domains. Especially, the heavy multi-reference frame search can be performed simultaneously and independently by using the proposed multi-frame linear motion predictor. In addition, all layers in STHME can be processed in parallel and suitable for hardware implementation. Future works can incorporate the proposed STHME algorithm into the hardware design and further verify its efficiency.

References

[1] *ISO/IEC ITU-T Rec. H264: Advanced Video Coding for Generic Audiovisual Services*, Joint Video Team (JVT) of ISO-IEC MPEG & ITU-T VCEG, Int. Standard, May 2003.

[2] JM 12.4, H.264/AVC JM Reference Software, Available: http://iphome.hhi.de/suehring/tml/.

[3] C. Kim, H. H. Shih, and C. C. J. Kuo, "Feature-Based Intra-Prediction Mode Decision for H.264," in *Proc. IEEE Int. Conf. Image Process.*, Singapore, Oct. 2004, vol. 2, pp.769-772.

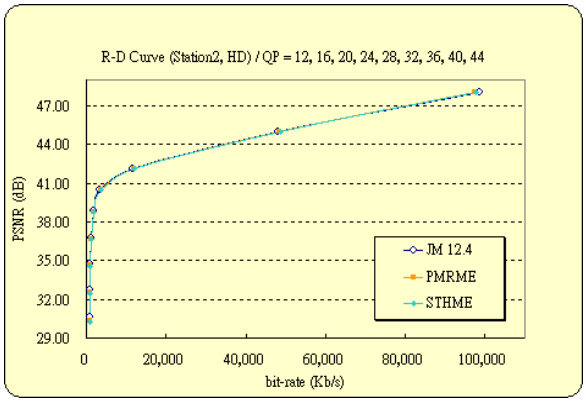
[4] Y. H. Chen, T. C. Chen, and L. G. Chen, "Hardware Oriented Content-Adaptive Fast Algorithm for Variable Block-Size Integer Motion Estimation in H.264," in *Proc. 2005 Int. Symp. Intell. Signal Process. and Commun. Syst.*, pp. 341-344.

[5] C. C. Lin, Y. K. Lin, and T. S. Chang, "PMRME: A Parallel Multi-Resolution Motion Estimation Algorithm and Architecture for HDTV Sized H.264 Video Coding," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Apr. 2007, vol. 2, pp.II-385 – II-388.

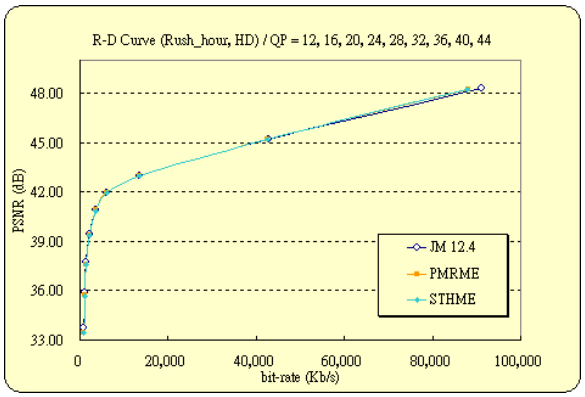
[6] Y. H. Hsiao, T. H. Lee, and P. C. Chang, "Short/Long-Term Motion Vector Prediction in Multi-Frame Video Coding System," in *Proc. IEEE Int. Conf. Image Process.*, Singapore, Oct. 2004, pp.1449-1452.

[7] L. M. Po and W. C. Ma, "A Novel Four-Step Search Algorithm for Fast Block Motion Estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 3, pp. 313-317, Jun. 1996.

[8] C. Hufnagl and A. Uhl, "Fast Block-Matching Algorithms for High-Resolution Video Compression," in *Proc. Int. Picture Coding Symp.*, Portland, Oregon USA, Apr. 1999, pp. 295-298.



(a)



(b)

Figure 5. The comparison of rate-distortion curves: (a) *station2*; (b) *rush_hour*.

TABLE IV. SIMULATION RESULT COMPARISONS (QP=12, 16, 20, 24, 28, 32, 36, 44)

Sequence	Algorithm	△PSNR$_y$	△bit rate	run time
station2	PMRME	-0.07	1.47%	4.56%
	STHME	-0.11	1.74%	1.99%
pedestrian area	PMRME	-0.09	7.81%	4.81%
	STHME	-0.10	9.38%	2.13%
rush hour	PMRME	-0.09	0.01%	3.07%
	STHME	-0.10	0.16%	1.31%