ADAPTIVE ALGORITHM FOR FAST MOTION ESTIMATION IN H.264/MPEG-4 AVC

Sergio Saponara, Massimiliano Melani, Luca Fanucci, Pierangelo Terreni

Department of Information Engineering, University of Pisa, via Caruso, I-56122 Pisa, Italy
fax: +39 050 2217522, email: {s.saponara, m.melani, l.fanucci, p.terreni}@iet.unipi.it

ABSTRACT

With reference to the H.264/MPEG-4 AVC video coding standard featuring interframe prediction with multiple reference frames and block sizes, this paper presents a motion estimator (ME) which exploits input signal variations to dynamically reconfigure the search size and the number of reference frames of an exhaustive block-matching search. This ME achieves, for bit-rates ranging from few Kbps to several Mbps, the same performance of the Full-Search with a considerable complexity reduction.

1. INTRODUCTION

H.264/MPEG-4 Advanced Video Codec (AVC) is the new video coding standard jointly developed by ITU-T and ISO/IEC [1,2]. While its basic framework is similar to the motion compensated hybrid scheme of previous standards, additional features improve the compression efficiency at the expenses of an increased implementation cost. The same visual quality of H.263 and MPEG-4 ASP can be achieved with bit-rate reductions up to 50% [1]. The main contribution to the increase of both coding efficiency and implementation complexity is due to the new features of interframe prediction: while in previous standards the motion estimation (ME) of the current frame refers to 1 reference picture and to 16x16-pixel image blocks, H.264/AVC considers multiple reference pictures and blocks sizes (16x16-pixel blocks and their partitions in 16x8, 8x16, 8x8, 8x4, 4x8 and 4x4 pixels).
Although the computation required by smaller block types can be reduced by considering block type overlapping in the computation data flow [3] (i.e. calculating the distortion of the smallest 4x4 block type and then adding corresponding subtype distortion to generate large bock types) the cost of ME in H.264/AVC is still prohibitive. Assuming, as in the standard C model [2], to have M=5 reference frames, a search range of ±p=16 and a Full Search (FS) algorithm then, for each image block, we need to examine $M\cdot(2p+1)^2 = 5445$ locations instead of $(2p+1)^2 = 1089$ as in previous standards. For each location an error cost function, usually the Sum of Absolute Differences (SAD), is evaluated: if *a(i,j)* and *b(i,j)* are the pixels of the current and reference blocks and *m, n* are the coordinates of the Motion Vector (MV) (i.e. position of the reference block within the search area), the SAD for a VxH-pixel block type is:

$$SAD(m,n) = \sum_{i=0}^{V-1}\sum_{j=0}^{H-1} \left| a(i,j) - b(i+m, j+n) \right|, (-p \le m, n \le p) \quad (1)$$

Therefore, the ME in H.264/AVC receives as input the current frame and (up to 5) multiple reference frames and produces as output the minimum SADs (SADmin) and relevant MVs for each 16x16-pixel image block and all its subpartitions in smaller blocks (down to 4x4). As an example, implementing the above algorithm for a 30 Hz CIF video (352x288 pixels) requires more than $16x10^9$ absolute difference and data loading operations per second at pixel level (8-bit data). These high computational and memory requirements (and the consequent high power consumption in custom hardware or software based realisations) represent a serious design challenge for resource-constrained electronic systems as wireless devices or cost-effective consumer terminals.
To address the above issues this paper proposes a ME which exploits input signal variations to dynamically reconfigure the search size and the number of reference frames of an exhaustive block-matching algorithm. Thus, unnecessary computations are avoided providing, for bit-rate applications ranging from few Kbps to several Mbps, the same compression efficiency as the FS with multiple reference frames and block sizes but with a noticeable complexity reduction. After this introduction Section 2 gives an overview of low-complexity ME techniques. Section 3 describes the proposed algorithm with reference to a variable block size, single reference frame scenario while Section 4 extends the technique to multiple reference frames. Experimental results, within the H.264 environment, are discussed and compared to the state of art. Conclusions are drawn in Section 5.

2. RELATED WORKS

Several ME techniques have been proposed in literature to reduce the complexity of the exhaustive search. Some of them (e.g. Three and Four Step Search, Log Search [4]) reduce the number of reference blocks investigated without exploiting the statistics of the input data. Instead of determining the global minimum over the search range usually a local minimum is obtained. On the contrary the class of predictive algorithms [4] exploits the spatial and temporal correlation of typical MV fields. The MV of a given block is predicted from a set of initial MV candidates, selected from its spatio-temporal neighbours, according to a certain law. To further reduce the estimation error a refinement process is then performed. For applications from tens to hundreds of Kbps the same high coding efficiency of the FS can be achieved with computational load reductions up to one order of magnitude. However, predictive algorithms are affected by two problems [5]: (i) the higher is the bit-rate of the considered application the

worse is the algorithm performance; (ii) the regularity of the exhaustive-search data flow is broken thus causing a poor data reuse, useful for reducing the number of external memory access, and a complexity increase of the data fetching. The limits of the above fast ME techniques can be overcome by adopting an exhaustive search which dynamically adjusts the search range according to the statistics of the input video signals [5,6]: the same estimation accuracy of the FS can be achieved for all bit-rates avoiding unnecessary computations and keeping a regular data flow. While [6] refers to the H.264 multiple block types scenario [5] is limited to a single block type/single reference frame scenario. Particularly, [6] proposes an exhaustive ME in which the search range for each block depends on the MVs of its spatial neighbouring blocks. The motion activity of a scene directly relates to the values of the relevant MV field. In case of high motion activity a large search window size permits to get a MV which results in small estimation error. When the amount of motion is small then searching over a large search area is not necessary. The performance of [6] in terms of coding efficiency and computational cost can be improved since it exploits spatial information only (not optimal for videos containing objects with different motion activities) and, in case of multiple reference frames, the search is simply repeated multiple times.

3. PROPOSED ME ALGORITHM

This section presents an adaptive motion estimator which overcomes the limits of [6] by: (i) exploiting both spatial and temporal correlation in the MV field; (ii) adopting the SADmin values as a measure of the motion prediction accuracy. In case of interframe coding, a high value of SADmin for the processed block indicates a high prediction error. Thus, by comparing the SADmin related to a certain block with proper thresholds the optimal search range for the successive blocks can be derived.

3.1 Adaptive Search Range Decision

The basic idea is to divide the analysis domain in 4 zones determined by comparing the minimum SAD related to a certain 16x16 image block (SADmin) with 3 thresholds T_1, T_2 and T_3 (with $T_3<T_2<T_1$). According to the resulting zone the search window size for the successive 16x16-pixel block and its subpartitions is differently determined by using the MVs of its spatio-temporal neighbouring blocks.

Zone 1 (T_1<SADmin): we are operating in high prediction error condition and hence, for the successive blocks, the search range is amplified (at least doubled as results from equation (3)) to permit getting larger MVs which result in small estimation errors. The new search range decision exploits both temporal and spatial correlation. A flag F is set to force the algorithm using spatio-temporal data. **Zone 2** (T_2<SADmin<T_1): if F is set the algorithm behaves likewise Zone 1, otherwise like Zone 3. F holds its value. **Zone 3** (T_3<SADmin<T_2): the prediction error has intermediate values, the analysis is based on spatial data. The flag F is reset.

Zone 4 (SADmin<T_3): the algorithm is working under optimal conditions and hence, for the successive blocks, the search range is kept small. In zone 4 we use only spatial data and we reset the flag F.

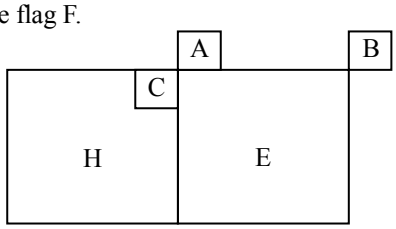

Fig. 1: Block location for search range decision

With reference to Fig. 1 E is the current 16x16-pixel block under motion prediction, H is the 16x16-pixel block previously encoded while A, B and C are spatial neighbouring blocks of size 4x4-pixel. Hereafter the MVs of A, B, C are referred as (MV_{AX}, MV_{AY}), (MV_{BX}, MV_{BY}), (MV_{CX}, MV_{CY}) while the MV that carries the temporal information is referred as (MV_{PX}, MV_{PY}). The maximum value that can be assumed by the search range is defined as *max_search_range*. The search range for E and its subpartitions is defined by the following steps (the same algorithm is applied separately to X and Y components allowing for search windows with rectangular shapes; hereafter we refer only to the X component):

Step 1. The SADmin of the block H is compared to T_1, T_2, T_3 to determine the operating zone.

Step 2. If the flag F=1 (we are in Zone 1 or 2) we assign to MV_{PX} the MV_X of the 16x16-pixel block occupying in the previous frame the same position of E. Otherwise MV_{PX} is set to 0 (thus using spatial correlation only).

Step 3. We determine the new search range according to equation (2) if we are in Zone 4 and to equation (3) otherwise.

$$search\_range_X = 1 + \max_{i=A,B,C}\left[abs\left(MV_{iX}\right)\right] \tag{2}$$

$$search\_range_X = \max\left\{abs\left(MV_{PX}\right), 2\cdot \max_{i=A,B,C}\left[abs\left(MV_{iX}\right)\right]\right\} \tag{3}$$

Step 4. We check if the $search\_range_X$ determined in Step 3 is lower than *max_search_range* (otherwise the new search range is *max_search_range*). Moreover, for video applications targeting a bit-rate greater than 500 Kbps we check if the $search\_range_X$ determined in Step 3 is higher than the parameter min_range_X defined as follows (otherwise min_range_X is the new search range):

$$\begin{aligned} &\min\_range_X = 2 \ \ if \sum_{i=A,B,C} abs\left(MV_{iX}\right) = 0 \\ &\min\_range_X = 3 \ \ if \ 0 < \sum_{i=A,B,C} abs\left(MV_{iX}\right) < 2 \\ &\min\_range_X = 4 \ \ otherwise \end{aligned} \tag{4}$$

When processing image borders, if in equations (2) and (3) a block (A, B or C) is not available the relevant MV components are set to (MV_{PX}, MV_{PY}).

3.2 Thresholds Analysis

The values for the thresholds T_1, T_2, T_3 adopted in Section 3.1 to evaluate the efficiency of motion prediction have been

derived by computer simulations of typical video sequences (with different grades of dynamism and image formats) as a good trade-off between the performance of the resulting motion estimator (see Sections 3.3 and 4.2) and the computational overhead of the range decision law. For the development and test of the proposed algorithm we used the following YUV 4:2:0 input videos: Mother&Daughter QCIF (176x144 pixels) (MD1) and CIF (MD2) and Foreman QCIF (FOR1) and CIF (FOR2) to test low and medium bit-rates applications; Mobile&Calendar SIF (352x240 pixels) (MC1) and CCIR (720x480 pixels) (MC2) and Stefan CIF (SF1) and CCIR (SF2) to test high bit-rates applications. The thresholds values depend on the quantization parameter (QP ranging from 1 to 51 in H.264) as expressed in (5).

$$T_1 = 75 \cdot QP + 2500, T_2 = 75 \cdot QP + 1500, T_3 = 35 \cdot QP + \alpha \quad (5)$$

being α=-100 when targeting bit-rates higher than 500 Kbps and α=900 when targeting bit-rates lower than 500 Kbps. As an example, with reference to the coding of 250 frames of SF1 (JM6.1 C model using the proposed ME with QP=24, 1 reference frame and a *max_search_range* of 16, first Intra frame followed by all Inter frames) Fig. 2 shows the statistics of the prediction error in each frame measured through the cost functions maxSAD, meanSAD and minSAD which are respectively the maximum, average and minimum of the SADmin values for all the blocks in a frame. Fig. 2 also reports the division of the analysis domain in the four zones described in Section 3.1. Similar results arise when analysing the other test videos.

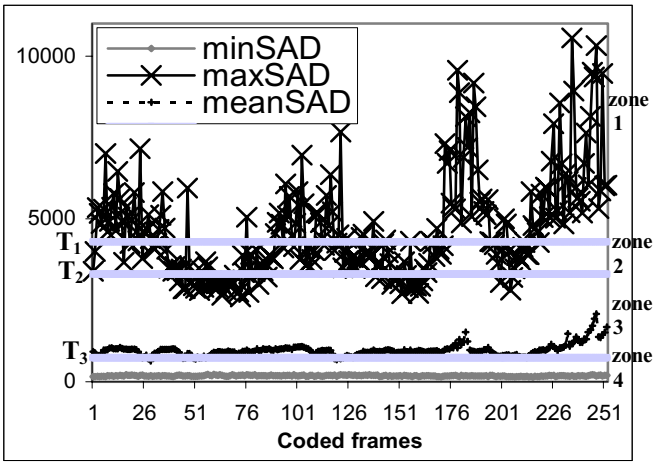

Fig. 2: SAD statistics and division in 4 zones for SF1

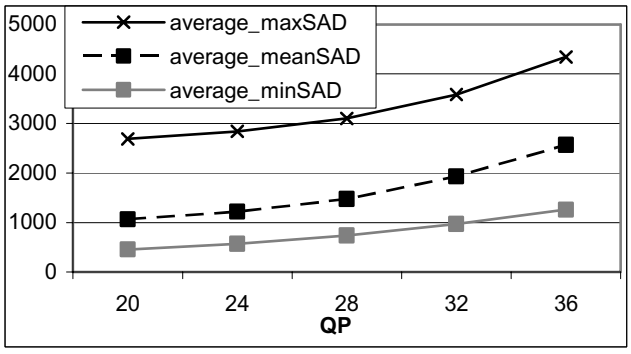

Fig. 3: SAD statistics vs. QP for MC1

As a matter of fact, the higher is the QP used by the encoder the higher are prediction errors and relevant SAD statistics: given a certain predicted field of MVs, high QP values lead to a high difference in the coding loop between the original frame and the reconstructed one. This is why we linked the values of T_1, T_2 and T_3 to the QP used in the coding loop according to the linear laws in (5). These dependencies have been extracted observing the variation of SAD statistics (minSAD, maxSAD, meanSAD) vs. QP for the considered tests. As an example, Fig. 3 shows for the MC1 video and five QP values the SAD statistics averaged over 250 frames.

3.3 Performance Results

This section compares the algorithmic performance and complexity of the proposed ME with respect to the FS and [6] approaches. The above analysis has been carried out using the JM6.1 C model of the H.264/AVC encoder [2]: the only piece of code modified was the ME. Processing speed results refer to a 2.4 GHz PentiumIV with 512 MB DDR SDRAM. All test videos (see Section 3.2) are coded for 10 sec considering a 30 Hz frame rate, 1 reference frame and a *max_search_range* of 16. Fig. 4 reports, for the three motion estimators and QP=24, the bit-rates achieved for equal PSNR values (39.8 dB for MD1, 38.6 dB for FOR1, 36.7 dB for MC1 and 38.3 dB for SF1). Fig. 5 shows the time (sec) spent for the ME task when encoding 300 frames.

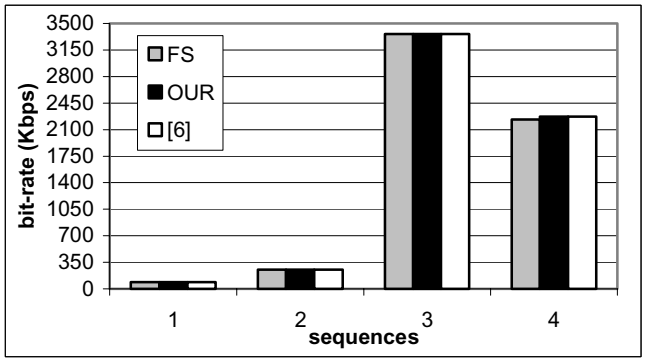

Fig. 4: Bitrate results for 1) MD1, 2) FOR1, 3) MC1, 4) SF1

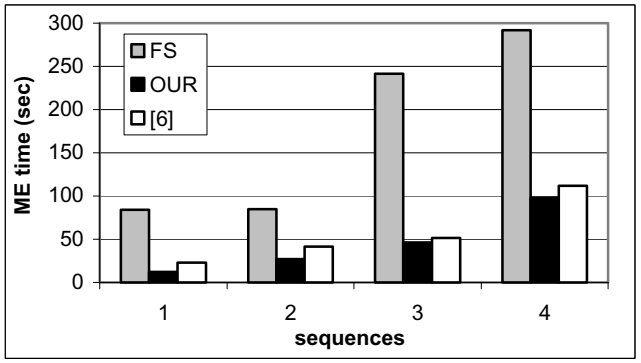

Fig 5: ME time for 1) MD1, 2) FOR1, 3) MC1, 4) SF1

From Fig. 4 we observe that the compression performances of the motion estimators are roughly the same over a wide range of bit-rates. The worst case occurs in Stefan for which we register a bit-rate increase of 2% with respect to the FS. For the other sequences the bit-rate increase is negligible. No

relevant differences arise with respect to [6]. The test in Fig. 4 has been repeated for several QP values (and hence PSNR values) leading to similar considerations. From Fig. 5 we observe that our algorithm features a speed increase ranging from a factor 2.9 up to 6.8 with respect to the FS and from 1.1 up to 1.9 with respect to [6]. Repeating the above analysis for MD2, FOR2, MC2 and SF2 leads to the same results achieved for MD1, FOR1, MC1, SF1.

4. MULTIPLE REFERENCE FRAMES ME

4.1 Adaptive Control of Reference Frames Number

In H.264/AVC multiple reference frames can be used to improve ME accuracy. In the standard C model [2] and in [6] the increase in computational complexity is directly proportional to the number of used reference frames since the basic searching algorithm is repeated for every reference frame. Sometime the gain in terms of coding efficiency is relevant, but in some cases is negligible, always requiring an increase of the computational burden. As proved in [3], when using H.264 with 5 reference frames with slow motion videos 90% of the optimal MVs refer to the first reference frame (thus multiple reference frames are not useful). In case of sequence with high dynamism this percentage is reduced to roughly 35%. We improved the basic ME proposed in Section 3 introducing a cost function to verify if, for all the blocks in the current frame under estimation, it is worth using multiple reference frames or not. The idea is to compare the maxSAD relevant to the previous encoded frame with a proper threshold T_4, if the prediction error maxSAD is lower than T_4 we will use only one reference frame otherwise we will use 5 reference frames. The value of T_4 depends on the QP according to the law reported in equation (6). It is to be noted that the values for T_4 and the adoption of maxSAD to control the number of used reference frames have been derived by computer simulations of typical video tests (see Section 3.2) as a good trade-off between the performance of the resulting motion estimator (see Section 4.2) and the computational overhead of the controlling law.

$$\begin{aligned} 1 \le QP < 16 &\rightarrow T_4 = 1700 \\ 16 \le QP < 28 &\rightarrow T_4 = 1700 + 50 \cdot (QP - 16) \\ 28 \le QP < 52 &\rightarrow T_4 = 2300 + 130 \cdot (QP - 28) \end{aligned} \quad (6)$$

4.2 Performance Results

Hereafter we repeat the analysis of Section 3.3 comparing the final ME algorithm described in Section 4.1 with the FS using 1 reference frame and 5 reference frames (respectively denoted as FS-1frame and FS-5frames in Figs. 6 and 7). In slow motion videos, as MD1 test, the introduction of multiple reference frames leads to a computational burden increase without noticeable benefits in terms of coding efficiency. Indeed, we measured a speed factor of 25 of our adaptive algorithm vs. the FS-5frames while (see Fig. 6) the PSNR and bit-rate results achieved for different QP values are the same. When repeating the analysis for videos characterised by complex motion fields (as MC1) the use of multiple reference frames leads to a sensible gain in performance. In Fig. 7 our adaptive algorithm and the FS-5frames behave similarly featuring a PSNR improvement of about 2 dB for all bit-rates with respect to FS-1frame.

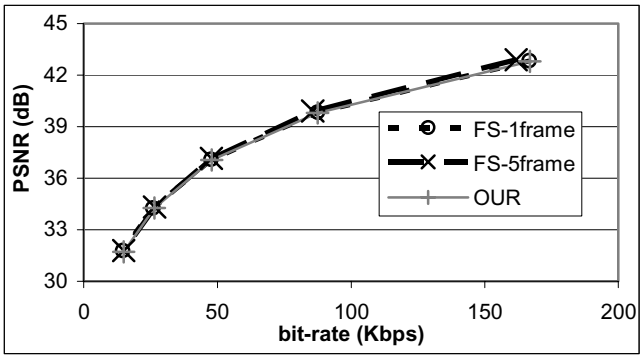

Fig. 6: Coding results with multiple reference frames, MD1

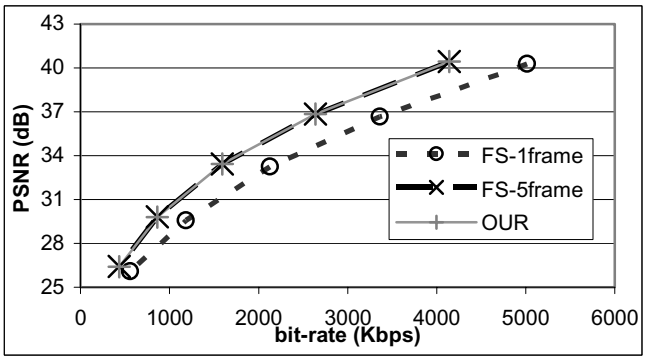

Fig. 7: Coding results with multiple reference frames, MC1

For MC1 we measured a speed factor of 2.2 of our adaptive algorithm vs. FS-5frames. Obviously, it is lower than the speed factor of 25 registered for the MD1 test because for the MC1 sequence the control law in Section 4.1 forces more frequently the ME to use multiple (5) reference frames.

5. CONCLUSIONS

This paper presents a motion estimator exploiting input signal variations to dynamically reconfigure the search area size and the number of reference frames of an exhaustive block-matching search. With reference to the H.264/AVC scenario, the same performances of the Full-Search are achieved with a complexity reduction from a factor 2 to 25 depending on the input video motion activity.

REFERENCES

[1] T. Wiegand et al., "Overview of the H.264/AVC video coding standard", *IEEE TCSVT*, 13 (7), pp. 560-576, July 2003

[2] ftp.imtc-files.org/jvt-experts/reference_software/jm6.1

[3] Y.W. Huang et al., "Analysis and reduction of reference frames for motion estimation in MPEG-4 AVC/JVT/H.264", *IEEE ICME03,* Baltimore, US, July 03, pp. 809–812

[4] P. Kuhn, *Algorithms, complexity analysis and VLSI architectures for MPEG-4 motion estimation*, Kluwer, 1999

[5] L. Fanucci et al., "VLSI design for low power data adaptive motion estimation", *EUSIPCO'02,* Toulouse, Sept.02, pp. 569–572

[6] M.C. Hong et al., "Further improvement of motion search range", JVT of ISO/IEC and ITU-T, C065, Fairfax, US, May 02