FAST MOTION ESTIMATION WITH SIZE-BASED PREDICTORS SELECTION HEXAGON SEARCH IN H.264/AVC ENCODING

Paolo De Pascalis[*], *Luca Pezzoni*[+], *Gian Antonio Mian*[*] *and Daniele Bagni*[+]

* Department of Information Engineering, University of Padova, Italy
\+ Advanced system Technology labs, STMicroelectronics, Agrate Brianza, Italy
e-mails: depa@dei.unipd.it luca.pezzoni@st.com

ABSTRACT

In this paper we present a new fast motion estimation algorithm suitable for H.264/AVC encoding systems. It applies a hexagon-based zonal search, thresholding criteria and improved predictors selection based on the size of the block under processing. The proposed method achieves almost the same visual quality of a Full Search block matcher with a dramatically reduced amount of computation.

1. INTRODUCTION

Motion Estimation (ME) is an important part of any video compression system, since it can achieve significant compression by exploiting the temporal redundancy existing in a video sequence. Unfortunately it is also the most computationally intensive function of the entire encoding process. In motion estimation the current image is divided into Macro-Blocks (MB) and for each MB, a similar one is chosen in a reference frame, minimizing a distortion measure. The best match found represents the predicted MB, while the displacement from the original MB to the best match gives the so-called Motion Vector (MV). Only the MV and the residual (i.e. the difference between the original MB and the predicted MB) need to be encoded and transmitted into the final stream. The distortion measure is the Sum of Absolute Differences:

$$SAD(d_x, d_y) = \sum_{m,n=0}^{N-1} \left| I_t(x+m, y+n) - I_{t-k}(x+d_x+m, y+d_y+n) \right|$$

where (d_x,d_y) represent the MV components and $I_t(x,y)$ the luminance value in frame t at coordinates (x,y). The search is carried out on a square Search Window (SW) of a predetermined Search Range (SR), so that there are $(2*SR+1)^2$ possible predicted macro-blocks and corresponding MVs.

Full Search Block-Matching (FSBM) motion estimation is the technique suggested in the reference software models of all the previous video coding standards, such as MPEG-1/2/4 and H.261/3. The FSBM algorithm exhaustively checks all the macro-blocks in the SW, thus finding always the optimum match, but it is the most computationally intensive ME algorithm possible and the most CPU-consuming part of the entire encoding process.

In the new, emerging H.264/AVC [1] standard, each 16x16 pixels MB can be sub-partitioned into smaller blocks, down to sizes of 4x4 pixels. This feature gives the ME process the ability to adapt to the local characteristics of the image, but it makes ME even more computationally intensive than in the case of other previous standards. Several fast ME algorithms have been proposed, but none takes advantage of the H.264/AVC enhanced partitioning system. It was proved [3,4] that ME using generalized predictors selection, zonal search and early termination criteria can achieve almost the same visual quality of the FSBM, while doing a dramatically reduced number of matches between blocks.

In this paper we present a new ME algorithm, suitable for H.264/AVC encoding schemes, using a hexagon-based zonal search, thresholding criteria and improved predictors selection based on the size of the block. The paper is organized as follows: Section 2 discusses about the proposed predictors selection criteria, Section 3 presents the modified hexagon pattern for prediction refinement. After some considerations on early termination criteria in Section 4, the algorithm is described in Section 5, followed by experimental results in Section 6 and conclusion in Section 7.

2. SIZE-BASED PREDICTORS SET SELECTION

In [3,4] was proposed a prediction set composed of the (0,0) vector, the median of the MVs of the left- up- and upright-blocks (respectively named A0, B0 and C0 in Fig. 1), the MVs of the 4 neighbouring blocks in the current frame (A0, B0, C0, D0 in Fig. 1), the ones of the co-located block (X1) and of the four vertically (B1, G1) and horizontally (A1, E1) adjacent blocks in the previous frame, and the acceleration MV (see Fig. 2).

In general we can state that the blocks correlated with the current one, which are likely to undergo the same motion, can be divided into three categories (see Fig. 1): spatially correlated blocks (A0, B0, C0, D0), neighbouring blocks in the previous frame (A1, B1, C1, D1, E1, F1, G1, H1) and co-located blocks in the previous two frames (X1 and X2), which provide the Acceleration MV (as shown in Fig. 2). This last one can enhance temporal prediction in sequences with fast and non-uniform motion.

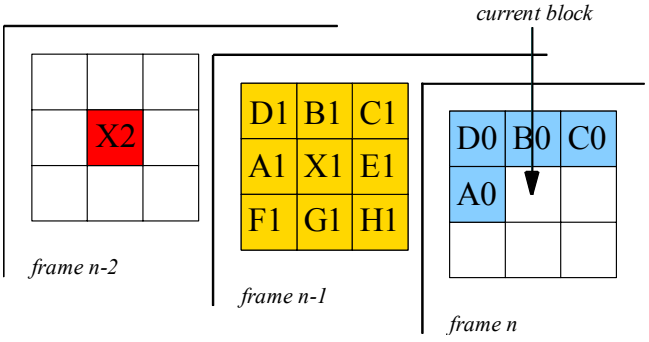

Fig. 1: Blocks correlated with the current one.

This work was carried out within the Italian Ministry of Education, University and Research (MIUR. Project "FIRB PRIMO").

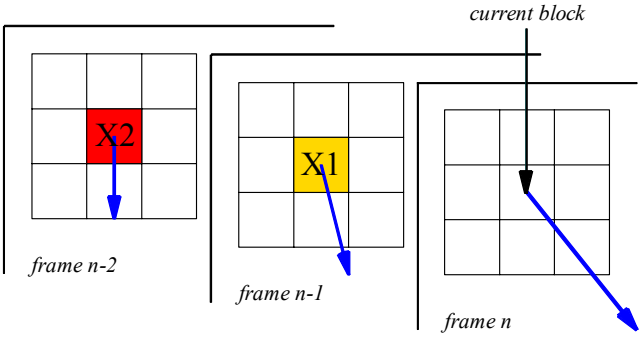

Fig. 2: Acceleration MV: $MV_{acc} = MV_{X2} + (MV_{X2} - MV_{X1})$

We experimented that using all these MVs in the prediction process was redundant, therefore in our method we selected a subset of them which ensures a good prediction for both slow and fast motion sequences: the (0,0) vector, the usual median predictor, the MVs of blocks A1, B1, X1 and D0 of Fig. 1 and the acceleration MV of Fig. 2.

While relying on this set of predictors for the 4x4 blocks, we propose a new predictor for all the blocks of size greater than 4x4: the mean of the MV of the smaller (e.g. 4x4) blocks belonging to the current block. Furthermore we use this new predictor as the only alternative to the median for all blocks grater than 4x4, in order to reduce the mean number of matches per block.

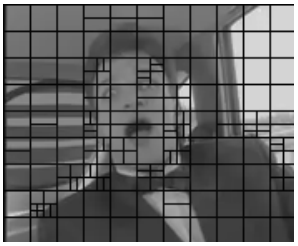

Fig. 3: Macro-Block partitioning for a frame of the Carphone sequence H.264/AVC encoded in QCIF format.

This idea is based on the consideration that the H.264/AVC codec chooses big blocks for uniform regions but it prefers small ones for detailed regions, as shown in Fig. 3. Assuming that the current block belongs to the background (or to a large uniform area), the MVs of the 4x4 blocks inside it will probably be very similar, and their mean value would be a good predictor, as the entire area undergoes the same motion. On the other hand, if the current block belongs to a detailed region, interested by a non-uniform motion, the mean predictor would not have much sense; but in such region a big block is very unlikely to be chosen by the encoder and having a bad prediction would not affect the whole performance of the compression system.

The choice of the mean instead of the median value is done to cope better with blocks containing two sub-blocks (e.g. 8x4 and 4x8 blocks), in which case the median would take into account only one of the two MVs. Moreover, the mean can be efficiently calculated with a logical right shift, being the number of 4x4 blocks always a power of 2, thus simplifying the hardware implementation.

3. HEXAGON-BASED ZONAL SEARCH

In [5] a hexagon pattern is proposed against the diamond of [2] and it is proven that the hexagon can often reach the final MV in less steps.

In our approach, we apply a modified internal pattern, instead of the 4-point pattern proposed in [5], for the final step of the search process. The new pattern, composed of 8 points (as shown in Fig. 4), is complementary to the hexagon and thus explores all the positions inside the last examined hexagonal area. The hexagon is used to find the local minimum around the predicted location: at each step it moves to the position of the current minimum, until the best match is found in its centre. Afterwards, all the positions inside the last hexagon are checked on the square pattern, and the final MV is selected.

With the diamond pattern, at each step (but the first) 3 or 5 positions need to be checked, depending on the location of the minimum on the previous diamond. With our approach, at each step (but the first) only 3 positions are checked as the other 3 belong to the previous hexagon; again this simplifies the hardware implementation.

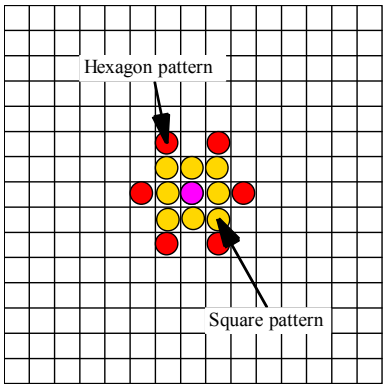

Fig. 4: The modified hexagon pattern.

4. EARLY TERMINATION

Sequences with low or global motion usually have more predictors close to the optimum and providing an acceptable distortion. To take advantage of this situation, early termination criteria can be applied to minimize the number of matches. In order to tune early termination to the local spatio-temporal characteristic of the video sequence, in [3,4] the calculation of a set of adaptive thresholds is proposed.

In our method we apply a unique adaptive threshold for all the predictors, in order to reduce the complexity of the algorithm and the computation overhead. If the SAD of a certain predictor is smaller than the threshold, then the search stops immediately, without checking any other location; otherwise the next predictor is checked, or (if the current predictor is the last one) the algorithm passes to the local search around the best match. The chosen threshold takes into account the minimum SAD found for the adjacent blocks and for the current block in the last frame:

$$T = \min(mSAD_{A0}, mSAD_{B0}, mSAD_{C0}, mSAD_{X1}) + npel$$

where $mSAD_i$ is the minimum SAD found for block i and *A0,B0* and *C0* refer to the left, up and upright block respectively, *X1* is the current block in the previous frame and *npel* is the number of pixels in the block (as shown in Fig. 1).

To increase the early termination probability, the predictors are checked in order of decreasing likelihood. While encoding the sequence, the algorithm takes count of the cases in which each predictor was chosen as the best one. At the beginning the first predictor is the median predictor, considered to be the most likely; then the predictors are sorted based on how many times each one gave the best match in the last N frames, where N is a fixed parameter.

5. SIZE-BASED PREDICTORS SELECTION HEXAGON SEARCH (SBPSHS)

The proposed algorithm [6] performs ME for the current MB, by examining blocks in increasing size order. The first step is the construction of the predictors set for the 4x4 blocks. If one of the predictors gives a SAD smaller than the threshold T, the search stops immediately, otherwise the predictor with the best match is chosen as the centre of the Search Window and the hexagon is placed on it.

After evaluating the SAD on all the locations of the hexagon (locations marked with '1' in Fig. 5), if the best match is found in the centre, the algorithm evaluates 8 more locations, the one of the square pattern placed inside the hexagon; otherwise the search pattern is moved on to the location of the best match found, and three additional matches are done (locations marked with '2' in Fig. 5). The algorithm continues evaluating the SAD on the moving hexagon, until the minimum value is found in the centre: at this point the square pattern is used to choose the final vector, which is then refined at ½ pixel and at ¼ pixel precision.

The pseudo-C code for the proposed SBPSHS algorithm is given in the following lines:

```
Start:
  if current block size is 4x4,
    goto Pred-A else goto Pred-B.
Pred-A:
  construct the predictors set composed of (0,0),
  spatial predictors (median predictor and the MVs
  of the up-left block) and temporal predictors
  (MVs of the collocated block , up-left and up
  blocks in the previous frame and the accelera-
  tion vector).
Pred-B:
  calculate the 2 predictors: median of left up
  and up-right block's MVs and mean of the MVs of
  the 4x4 blocks inside current block.
SetT:
  compute threshold T as the minimum of the best
  SAD found for the co-located block in the previ-
  ous frame and for the left up and up-right
  blocks in the current frame plus the number of
  pixels in the current block.
Choose-Pred:
  for each predictor in the predictors set do:
    current_SAD=SAD at the current_pred
    predictor's location.
  if (current_SAD<T)
    h_center=current_pred and goto Hexagon,
  else evaluate next predictor.
Hexagon:
  construct the hexagon pattern around h_center;
  evaluate the SAD for all locations on the hexa-
  gon not previously checked.
  If minSAD is found in h_center go to Square,
  else h_center=position of the current minimum
  and goto Hexagon
Square:
  build the square pattern around h_center; evalu-
  ate the SAD for all the locations of the square
  and found the minimum SAD.
End:
  the final MV is the location where the minimum
SAD was found.
```

Fig. 5 reports an example of local minimum search: the algorithm performs 3 steps in the hexagon search starting at the location of the best predictor. The location of the minimum at step 3 is the same found at step 2, therefore the hexagon search stops and the square pattern is used to choose the final MV.

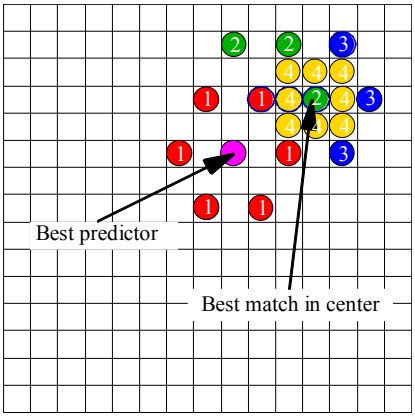

Fig. 5: Hexagon search: the numbers on the locations correspond to the step they belong to. The search starts at the predicted location and takes 3 hexagon steps.

6. RESULTS

We present the experimental results with the proposed algorithm implemented into the encoder reference model JM 7.3.

In Table 1 a comparison between SBPSHS and FSBM is reported for QCIF and CIF sequences compressed at QP=31, Intra period=12 and two different SR values (16 and 32). As the tests were made at constant QP, the most important result is the length of the encoded bitstream (in Bytes/Picture), which indicates the compression rate achieved for the given visual quality QP parameter. The 6th column reports the per cent increase of bytes-per-picture for SBPSHS against FSBM.

Table 1 also shows the average Peak Signal-to-Noise Ratio (PSNR) and the number of matches performed per block (NM column). For FSBM the latter is a constant value equal to $(2*SR+1)^2$, whereas for SBPSHS it is an average value and it is determined by the number of steps required in the hexagon search, and by the thresholding criterion. Note that in SBPSHS the average number of matches per block is almost the same for SR=16 and for SR=32.

Seq.	SR		PSNR	B/Pict	Incr%	NM	SF
foreman	16	FSBM	33.93	463.70		1089	
QCIF	16	SBPSHS	33.90	473.96	2.21	10.3	106
QP=31	32	FSBM	33.94	463.58		4225	
	32	SBPSHS	33.91	472.91	2.01	10.3	410
teeny	16	FSBM	33.77	1315.25		1089	
QCIF	16	SBPSHS	33.74	1331.19	1.21	10	109
QP=31	32	FSBM	33.77	1318.63		4225	
	32	SBPSHS	33.75	1326.04	0.56	10.8	391
pingpong	16	FSBM	32.92	3059.89		1089	
CIF	16	SBPSHS	32.88	3166.41	3.48	10.1	108
QP=31	32	FSBM	32.92	3045.36		4225	
	32	SBPSHS	32.89	3147.81	3.36	10.7	395
mobile	16	FSBM	31.63	6537.91		1089	
CIF	16	SBPSHS	31.63	6547.04	0.14	10.3	106
QP=31	32	FSBM	31.63	6554.27		4225	
	32	SBPSHS	31.63	6548.62	-0.09	10.3	410

Tab 1: comparison between SBPSHS and FSBM: SR=Search Range, B/Pict= average Bytes per picture, NM=average number of matches per block , SF=Speedup Factor**.**

Seq.	SR		PSNR	Incr%	NM	SF
foreman	16	FSBM	31.03		1089	
QCIF	16	SBPSHS	30.89	-0.46	10.6	103
64 kbit/s	32	FSBM	31.02		4225	
	32	SBPSHS	30.84	-0.59	10.6	399
teeny	16	FSBM	25.21		1089	
QCIF	16	SBPSHS	25.07	-0.53	9.7	112
64 kbit/s	32	FSBM	25.16		4225	
	32	SBPSHS	25.11	-0.20	10.1	418

Tab 2: comparison between SBPSHS and FSBM at low bit rates: SR=Search Range, NM=average number of matches per block, SF=Speedup Factor.

Seq.	kbit/s		PSNR	Incr%	NM	SF
pingpong	768	FSBM	33.89		1089	
CIF SR=16	768	SBPSHS	33.69	-0.58	9.98	108
	1024	FSBM	35.33		1089	
	1024	SBPSHS	35.19	-0.41	9.98	109
mobile	768	FSBM	29.47		1089	
CIF SR=16	768	SBPSHS	29.45	-0.07	10.1	108
	1024	FSBM	30.78		1089	
	1024	SBPSHS	30.75	-0.10	10	109

Tab 3: comparison between SBPSHS and FSBM at high bit rates: SR=Search Range, NM=average number of matches per block, SF=Speedup Factor.

The speedup factor (SF column) is computed as the ratio between the average number of matches per block of the two algorithms.

Tables 2 and 3 present a comparison between the two algorithms made at constant bit rate. Table 2 refers to QCIF sequences encoded at 64 kbit/s, while Table 3 to CIF sequences at high bit rates (768 kbit/s and 1 Mbit/s).

As it can be seen in the three tables, SBPSHS ensures a PSNR very close to the one of FSBM at both high and low bit rates, and a speedup factor around 100 with Search Range of 16 pixels, and around 400 with Search Range of 32 pixels.

7. CONCLUSIONS

We have presented a new, fast motion estimation technique [6] suitable for H.264/AVC compression schemes. The size-based prediction technique proved to be efficient on QCIF/CIF videoconferencing sequences in both terms of quality (PSNR) and speed. The gain in speed is given by the low average number of matches per Macro-Block, which results in a speedup factor around 400 against the FSBM algorithm with a Search Range of 32 pixels.

The hexagon pattern proved to be efficient in giving robustness against local minima, while the enhanced predictors set gave a good estimate allowing early termination criteria to maintain a low average number of matches per block.

Future work will focus on multiple-stage ME using the hexagon pattern and the predictors selection criterion presented here, for SD-TV and HD-TV compression.

REFERENCES

[1] T. Wiegand, G. J. Sullivan, "Overview of the H.264/AVC video coding standard", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, n. 7, July 2003.

[2] S. Zhu and KK Ma, "A New Diamond Search Algorithm for Fast Block-Matching Motion Estimation", *IEEE Transactions on Image Processing,* vol.9, no.2, February 2000.

[3] A. M. Tourapis, O. C. Au and M. L. Liou, "Highly Efficient Predictive Zonal Algorithms for Fast Block.-Matching Motion Estimation", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, n. 10, pp 934-947, October 2002.

[4] H-Y C. Tourapis, A. M. Tourapis, "Fast Motion Estimation within the JVT codec", *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG 5th meeting*, Geneva Switzerland, 09-17 October 2002.

[5] C. Zhu, X. Lin and L.P Chau, "Hexagon-Based Search Pattern for Fast Block Motion Estimation", *IEEE Transactions on Circuits and Systems for Video Technology,* vol.12, no.5, pp 349-355, May 2002.

[6] P. De Pascalis, "Stima del Moto per Codificatori H.264", *Laurea Degree Thesis, Dept. of Information Engineering, University of Padova*, Italy, October 2003.