Low-complexity motion estimation for the Scalable Video Coding extension of H.264/AVC

Livio Lima[a], Daniele Alfonso[b], Luca Pezzoni[b], Riccardo Leonardi[a]

[a]Department of Electronics for Automation, University of Brescia, Brescia, Italy
[b]Advanced System Technology Labs, STMicroelectronics, Agrate Brianza, Italy

ABSTRACT

The recently standardized Scalable Video Coding(SVC) extension of H.264/AVC allows bitstream scalability with improved rate-distortion efficiency with respect to the classical Simulcasting approach, at the cost of an increased computational complexity of the encoding process. So one critical issue related to practical deployment of SVC is the complexity reduction, fundamental to use it in consumer applications. In this paper, we present a fully scalable fast motion estimation algorithm that enables an excellent complexity performance.

Keywords: Fast Motion Estimation, H.264, Scalable Video Coding

1. INTRODUCTION

Most of the activity of the ISO and ITU Joint Video Team (JVT) over the last few years has been dedicated to scalable video, and this work has recently seen recognition in the so called "Scalable Video Coding"extension(SVC) of the H.264/AVC standard for video compression[1,2] . Contrasting from the classical video coding approach, the scalable paradigm enables the decoding from a unique coded representation(bitstream) at different "working points"in terms of spatial, quality and temporal resolution. The main drawback of the SVC architecture, shown in Figure 1, is the complexity increase compared to H.264 single layer coding. In SVC the original video sequence is downsampled to generate lower spatial resolutions that can be encoded at different quality layers. The lowest decodable point (in terms of spatial and quality resolution) is called Base Layer and is H.264/AVC compatible, while the others layers are called enhancement layers. The Inter-layer prediction is a new tool introduced in SVC that enables the reuse of the motion, texture and residual information from lower layers to improve the compression efficiency of the enhancement layers. In particular, from the motion estimation point of view, it has been shown that better compression performance are obtained by performing the full motion estimation process independently at each layer, where for the enhancement layers additional new macroblock modes(introduced by the Inter-layer prediction and defined in SVC standard) have to be evaluated. Because the motion estimation process is responsible for most of the encoding time, it is clear as this multi-layer architecture drastically increases the complexity compared to single-layer coding. This is one of the reason why the success of this scalable video coding extension will depend on the tradeoff between complexity and performance compared to the use of simulcast or transcoding solutions. A complexity analysis of the new SVC standard con be found in[3] .

This work presents the full scalable extension of a fast motion estimation algorithm for the base layer and temporal scalability that was presented in[4] . The overall proposed algorithm not only decreases the complexity of the motion estimation process for the enhancement layers (independently from the adopted scalability configuration), but it also provides a fast motion estimation algorithm for the base layer. This is the reason why different algorithms are used for motion estimation in the base layer and in the enhancement layers, as will be described in the following. The results show that the proposed algorithm could greatly decrease the complexity in terms of number of tested motion vectors with comparable compression performance to the fast motion estimation algorithm proposed in the reference software[5] .

Further author information: (Send personal correspondence to Livio Lima or Daniele Alfonso)
Livio Lima: E-mail: livio.lima@ing.unibs.it, Telephone: +390303715457
Daniele Alfonso: E-mail: daniele.alfonso@st.com, Telephone: +3902

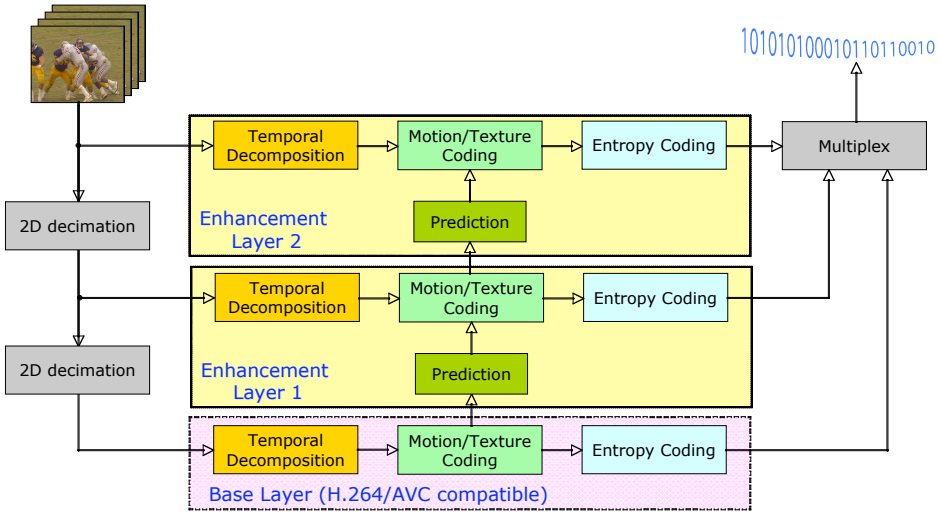


Figure 1. Scalable Video Coder structure

The remainder of the paper is structured as follow. Sections 2 gives a brief explanation on how the proposed algorithm works for the base layer, while in Section 3 the multi-layer extension is proposed. Finally Section 4 provides the conducted experimental simulations.

2. MOTION ESTIMATION IN BASE LAYER

The motion estimation algorithm in SVC base layer is based on two main steps: the Coarse Search and the Fine Search. The Coarse search is a pre-analysis step useful to initialize the Fine Search, which provides the motion vectors that will be used to actually encode each block.

2.1 Coarse Search

The Coarse Search is a pre-processing step that finds a single motion vector for each 16x16 macroblock of each frame following the display order and it uses only the previous original frame as reference. The Coarse Search could be applied on the whole sequence before the encoding process or independently within each Group of Picture (described in the follow). If the current macroblock is at position (i, j) in frame n, the Coarse Search tests 3 spatial predictors and 3 temporal predictors(obviously available from the second frame), where the 6 predictors are the motion vectors already computed for the Coarse Search of previous macroblocks. The spatial predictors are the vectors of the macroblocks in position $(i-1, j)$, $(i, j-1)$, $(i-1, j-1)$ in frame n, while the temporal predictors are related to the macroblocks in position (i, j), $(i, j-1)$, $(i-1, j)$ in frame n-1. Subsequently a grid of 12 fixed motion vectors called "short updates"at half pel accuracy are added to the best spatial/temporal predictor to get the best motion vector for the 16x16 macroblock. At each step the criteria for the choice of the best motion vector is the minimization of the Sum of Absolute Differences (SAD). The vectors estimated during the Coarse Search do not have coding purposes, but are used as a good starting point for the Fine Search step explained in the next section.

It is important to note as the Coarse Search process is performed only on the input spatial resolution used to generate the base layer, that is potentially a downsampled version of the video sequence used to encode the enhancement layers, as in case of spatial scalability. It follows as the motion information generated by the Coarse Search has to be adjusted in order to be used in the Fine Search of the enhancement layers, as will be explained in section 3.

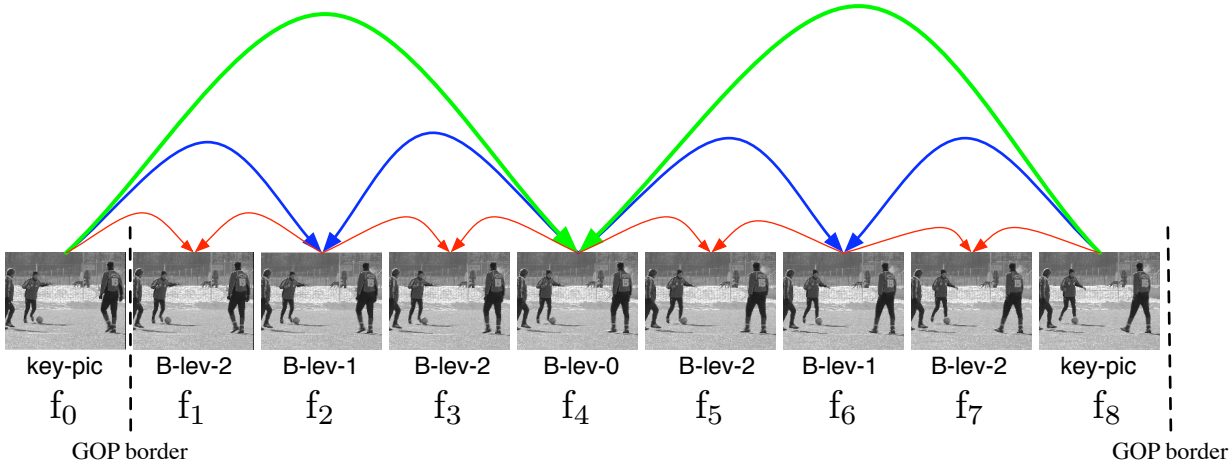


Figure 2. Hierarchical B-frame decomposition structure for a single GOP

2.2 Fine Search

The Fine Search is the step of the algorithm that is responsible for the estimation of the final motion vectors for each macro-blocks that are subsequently used for the motion compensation and coding. The Fine Search is applied on each frame following the encoding order given by the particular temporal decomposition structure. In the rest of this work we considered only the Hierarchical B-frame decomposition, which enables a native temporal scalability with improved performance compared to other structures[6] . As shown in Figure 2, the Hierarchical B-frame decomposition processes the video sequence in Group of Pictures (GOP) where for each GOP the last frame, called key-picture, is intra-coded(I-frame) or inter-coded(P-frame) with the previous key-picture as reference. Al the other pictures within the GOP are inter-coded as bidirectional pictures(B-frame) using the reference as shown in Figure 2. Since Hierarchical B-frame enbles a closed-loop motion estimation, during the Fine Search the motion estimation is performed using the decoded version of the reference frames.

For the understanding of the proposed algorithm, it is also important to note that the bidirectional motion-estimation (for B-frames) for each block is not performed by joint search of the forward and backward motion vectors. First the best forward and backward vectors are independently estimated by one-directional motion estimation, then an iterative procedure "corrects"the vectors for bi-directional estimation. At each step of the iterative procedure one motion vector is fixed while the other one is refined. For this reason the Fine Search is further split in 2 steps: one-directional step followed by a bi-directional refinement. In B frames the one-directional step is applied twice to search the best forward and backward motion vectors. The bi-directional refinement is then applied. P frames instead need only the one-directional step to find the best backward motion vector.

For each macroblock the one-directional step is applied for each block type because different partitions are evaluated for each macroblock. This means that the motion estimation process has to be performed for each possible sub-block(16x16, 16x8, 8x16, ...). Similarly to the Coarse Search, the Fine Search tests 3 temporal predictors and 3 spatial predictors, where the difference is in the meaning of the temporal and spatial predictors. In fact the spatial predictors are the result of the Fine Search already performed for spatially adjacent blocks of the same size while the temporal predictors are the results of the Coarse Search scaled by an appropriate ratio, as shown in Figure 3(a).

To understand the meaning of the temporal and spatial predictors, let us consider the following example. Suppose to apply the one-directional step for the macroblock (i, j) of the frame f_4, inspecting the macroblock mode M_x, in order to obtain the Fine Search vectors $\mathbf{f}_{4,b}(i, j, M_x)$ and $\mathbf{f}_{4,f}(i, j, M_x)$. Let us assume that $\mathbf{c}_4(i, j)$ is the Coarse Search motion vector for the macroblock (i, j) in frame f_4 that has been estimated with a motion estimation performed with respect to the frame f_3, as presented in section 2.1 (in the Coarse Search the motion

estimation is performed with respect to the previous frame). Since the temporal distance between f_4 and its references (f_0 and f_8) is equal to 4 pictures, the temporal predictors are rescaled by a factor of 4. The sets of temporal(T) and spatial(S) predictors for backward and forward motion vectors are given by:

$$T_b(i,j) = \{4\mathbf{c}_4(i,j), 4\mathbf{c}_4(i-1,j), 4\mathbf{c}_4(i,j-1)\} \qquad T_f(i,j) = \{-4\mathbf{c}_4(i,j), -4\mathbf{c}_4(i-1,j), -4\mathbf{c}_4(i,j-1)\}$$

$$S_b(i,i) = \{\mathbf{f}_{4,b}(i-1,j-1,M_x), \mathbf{f}_{4,b}(i-1,j,M_x), \mathbf{f}_{4,b}(i,j-1,M_x)\}$$

$$S_f(i,i) = \{\mathbf{f}_{4,f}(i-1,j-1,M_x), \mathbf{f}_{4,f}(i-1,j,M_x), \mathbf{f}_{4,f}(i,j-1,M_x)\}$$

The best backward and forward predictors are chosen through a RD-optimization

$$\mathbf{p}_b(i,j,M_x) = \arg\min_{\mathbf{x}\in T_b,S_b}(d(\mathbf{x}) + \lambda_{mot}\cdot r(\mathbf{x}, bl\_mode\_M_x))$$

$$\mathbf{p}_f(i,j,M_x) = \arg\min_{\mathbf{x}\in T_f,S_f}(d(\mathbf{x}) + \lambda_{mot}\cdot r(\mathbf{x}, bl\_mode\_M_x))$$

where $d()$ is the MSE on the block (i,j) obtained using the vector $\mathbf{x}$ and $r()$ is the rate associated to $\mathbf{x}$ and the choice of the mode M_x.

The best predictor is then refined through 3 different sets of update vectors: short(U_S), medium(U_M) and long(U_L), where the new groups of medium and long updates are defined in order to take in account the distance between current and reference frame in case of possibly long GOPs. In particular, if $D \geq 8$ long, medium and short updates are tested; if $D = 4$ or $D = 2$ medium and short updates are tested and if $D = 1$ only short updates are tested. So, for the above example, the best "updated backward predictor"($\mathbf{u}_b$) is given by:

$$\mathbf{u}_b^M(i,j) = \mathbf{p}_b + \arg\min_{\mathbf{u}\in U_M}(d(\mathbf{p}_b+\mathbf{u}) + \lambda_{mot}\cdot r(\mathbf{p}_b+\mathbf{u}, bl\_mode\_M_x))$$

$$\mathbf{u}_b(i,j) = \mathbf{u}_b^M + \arg\min_{\mathbf{u}\in U_S}(d(\mathbf{u}_b^M+\mathbf{u}) + \lambda_{mot}\cdot r(\mathbf{u}_b^M+\mathbf{u}, bl\_mode\_M_x))$$

The number and the values of the updates, as well as the threshold value D are experimentally derived through an extensive set of simulations over different test sequences with different coding parameters in order to obtain the best tradeoff between performance and complexity. After the updates evaluation, for efficiency purpose $\mathbf{u}_b$ is compared to the zero motion vector($\mathbf{z}$) and the H.264 predictor($\mathbf{p}_{264}$) and the best one is finally refined at quarter-pel accuracy(with 9 vectors taken from the set U_{QP} and that have each component equal to -1, 0 or +1), in order to find the final Fine Search motion vector $\mathbf{f}_{4,b}(i,j,M_x)$ for the MB mode M_x:

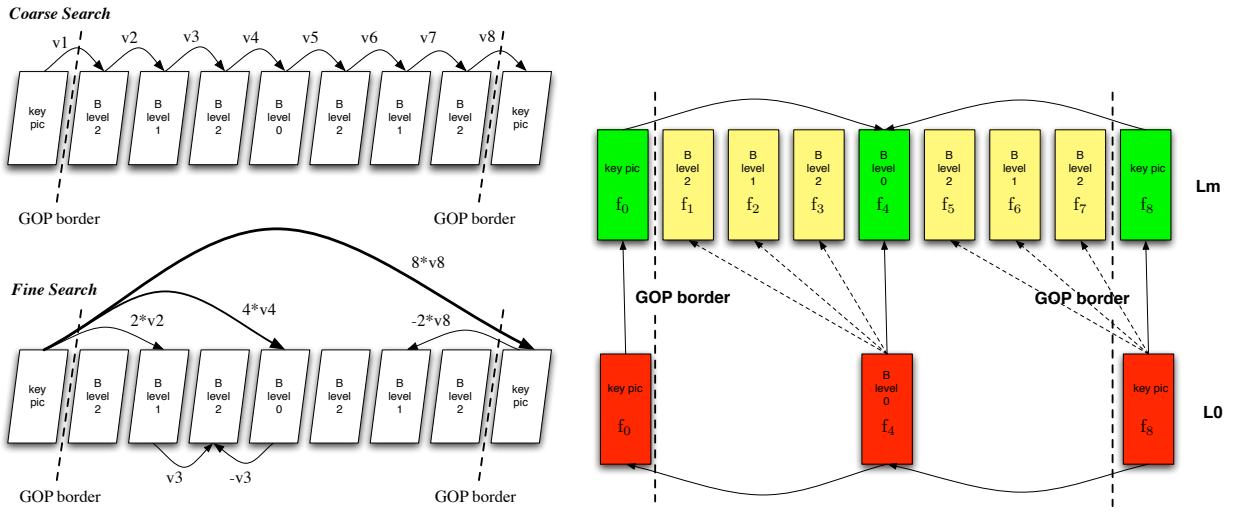
$$\hat{\mathbf{f}}_{4,b}(i,j,M_x) = \arg\min_{\mathbf{x}\in\{\mathbf{u}_b,\mathbf{z},\mathbf{p}_{264}\}}(d(\mathbf{x}) + \lambda_{mot}\cdot r(\mathbf{x}, bl\_mode\_M_x))$$

$$\mathbf{f}_{4,b}(i,j,M_x) = \hat{\mathbf{f}}_{4,b} + \arg\min_{\mathbf{u}\in U_{QP}}(d(\hat{\mathbf{f}}_{4,b}+\mathbf{u}) + \lambda_{mot}\cdot r(\hat{\mathbf{f}}_{4,b}+\mathbf{u}, bl\_mode\_M_x))$$

The bi-directional motion vectors are obtained through an iterative refinement of one-directional vectors. At each step of the iterative procedure, one motion vector is fixed while the other one is refined through 8 updates at quarter pel accuracy.

3. MULTI LAYER EXTENSION

To simplify the notation, referred to Figure 3(b) the base layer (BL) is identified by L_0 while a general enhancement layer (EL) is represented as L_m. Exploiting the motion information from lower layers, for each picture of higher layers we can expect to have a good representation of the motion using an appropriate scaled version of the motion flow of the corresponding pictures at lower layers. This is not true when a particular picture in a higher level has no associated picture at lower layers, for example when a different frame rate is used from one layer to another, thus a different motion estimation approach is used for pictures with or without an associated frame in lower layers. This problem is shown in Figure 3(b), where an 8-picture GOP is considered and the EL has a frame-rate 4 times larger then the BL. In this case, for the key-pictures (if of P-type) and for B-level-0



(a) (b)

Figure 3. 3(a): Scaling of Coarse Search motion vectors to obtain temporal predictors in Fine Search; 3(b): Motion scaling between layers with different frame-rate

pictures the motion information is directly inferred from the corresponding pictures in the BL, following the process described in section 3.2. This case leads to a better compression efficiency and a speed-up of the motion estimation process. For the B-level-1 and -2 pictures the correspondence with the BL is missing, and consequently the motion information is similarly extracted as was done for the BL (see subsection 3.1).

The problem of the different frame-rate is not the only aspect that has to be considered during the multi-layer extension process. In fact the SVC standard allows a particular type of spatial scalability, named Extended Spatial Scalability (ESS), where generally the BL is a scaled and cropped version of EL, as in case of SDTV to HDTV scalability, for which SDTV represents a base layer with 4:3 aspect ratio whereas HDTV corresponds to a 16:9 aspect ratio enhancement layer. The ESS defines the concept of Cropping Window (CW), that is the area of the EL used to generate the BL, as shown in Figure 4. In sections 3.1 and 3.2 we will refer as (W_{BL}, H_{BL}) the dimension of the BL, (W_{EL}, H_{EL}) the dimension of the EL, (x_{CW}, y_{CW}) the origin of the CW inside the EL and with (W_{CW}, H_{CW}) the dimension of the CW. Obviously, all such parameters affect the scalability configuration. So, for eample, if $(x_{CW}, y_{CW}) = (0, 0)$ and $(W_{CW}, H_{CW}) = (W_{EL}, H_{EL}) = (2W_{BL}, 2H_{BL})$ we assume dyadic spatial scalability; if $(x_{CW}, y_{CW}) = (0, 0)$ and $(W_{CW}, H_{CW}) = (W_{EL}, H_{EL}) = (W_{BL}, H_{BL})$ we will have CGS, and so on. Therefore an enhancement layer can be of any type: CGS, MGS, dyadic spatial or ESS.

3.1 Frame without an associated match in lower layers

When the lower layers do not provide any motion information to the upper ones, the motion estimation process for the EL follows the algorithm described in section 2 for the BL. The only difference concerns the Coarse Search, since as previously mentioned the full Coarse Search process is performed only for the BL. In order to determine the temporal predictors for the Fine Search at higher layers, a scaling of the motion vectors obtained from the Coarse Search is performed. Hereafter, the process is explained only for one EL with respect to the BL. It can be easily extended in a similar way between 2 consecutive enhancement layers. Let us define the frame rate ratio $f_R = f_{EL}/f_{BL}$ as the ratio between the frame rates of the EL(f_{EL}) and BL(f_{BL}), and the resolution ratios as follow:

$$r_X = \frac{W_{CW}}{W_{BL}} \qquad r_Y = \frac{H_{CW}}{H_{BL}} \tag{1}$$

Referring to Figure 3(b), let us estimate the temporal predictor $\mathbf{c}_{2,EL}(i, j)$ for the macroblock (i, j) at position $[x_{EL}(i, j), y_{EL}(i, j)]$ in frame f_2 (and similarly for f_1, f_3) of the EL. The Coarse search motion vector $\mathbf{c}_{4,BL}(h, k)$

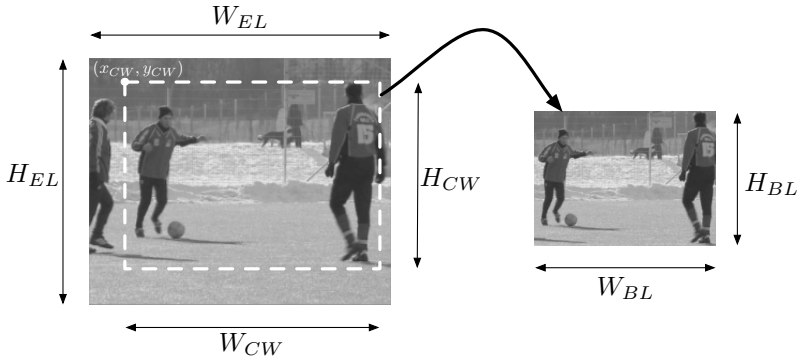


Figure 4. Base layer generation from Cropping Window

estimated for frame f_4 of the BL has to be used to infer the temporal predictors. If the current macroblock lies inside the cropping window, the Coarse Search motion vectors $\mathbf{c}_{2,EL}(i,j)$ for MB (i,j) in the EL is computed as

$$\mathbf{c}_{2,EL}(i,j).x = \frac{\mathbf{c}_{4,BL}(h,k).x * r_X}{f_R} \qquad \mathbf{c}_{2,EL}(i,j).y = \frac{\mathbf{c}_{4,BL}(h,k).y * r_Y}{f_R} \tag{2}$$

where h and k are the indexes of the macroblock of the BL with coordinates $[x_{BL}(h,k), y_{BL}(h,k)]$ for the upper-left pixel given by:

$$x_{BL}(h,k) = \frac{x_{EL}(i,j) - x_{CW}}{r_X} \qquad y_{BL}(h,k) = \frac{y_{EL}(i,j) - y_{CW}}{r_Y} \tag{3}$$

If the MB (i,j) lies outside the cropping window, as in the case of ESS with cropped BL, we can not use the BL motion information, and so $\mathbf{c}_{2,EL}(i,j)$ is set to $\mathbf{0}$. After this scaling process, the Fine Search is performed as explained in section 2.2. So, the set of temporal predictors for frame f_2 in the EL, is given by:

$$T_b(i,j) = \{2\mathbf{c}_{2,EL}(i,j), 2\mathbf{c}_{2,EL}(i,j-1), \mathbf{c}_{2,EL}(i-1,j)\}$$
$$T_f(i,j) = \{-2\mathbf{c}_{2,EL}(i,j), -2\mathbf{c}_{2,EL}(i,j-1), -2\mathbf{c}_{2,EL}(i-1,j)\}$$

3.2 Frame with match in lower layers

For the pictures with corresponding low-layer representations, like the KP and B0 pictures in Figure 3(b), we can fully exploit the motion information of the BL, expecting good performance with reduced computational complexity. However, these considerations are not completely true in case of ESS scalability so, in general, the reuse of the BL motion information can be done only for the blocks of the EL that lie within the cropping window. Furthermore, the performance depends also on the quality of the pictures in the BL. The higher the quality of the BL the more efficient the inter-layer prediction, both for texture and motion information. In order to show the dependencies between the quality of th BL and the performance of the proposed algorithm, two scenarios are considered:

- low complexity: the one-directional step of the Fine Search tests only 1 inter-layer predictor inferred from the lower layers (see below), together with the predicted motion vector provided by the SVC encoder and the zero motion vector. The best vector is finally refined through 8 updates at quarter pel accuracy. The bi-directional step is the same of the BL.
- high complexity: the one-directional step of the Fine Search tests 1 inter-layer predictor inferred from lower layers and as explained in section 2.2 refines it with short, medium and long updates, together with the predicted motion vector provided by the SVC encoder and the zero motion vector. The best vector is finally refined through 8 updates at quarter-pel accuracy. The bi-directional step is the same of the BL.

It's important to note that in this case the Fine Search does not test the temporal predictors; as such the scaling of the Coarse Search information, as described in section 3.1, is not needed.

Again, referring to Figure 3(b), let us consider the one-directional inter-layer predictor (backward or forward) $\mathbf{p}_4(i,j,M_x)$ for the macroblock (i,j) at position $[x_{EL}(i,j), y_{EL}(i,j)]$ (which lies inside the Crop Window) in frame f_4 of the EL for the particular macroblock mode M_x. The block type associated with the macroblock mode M_x has a relative position (x_{M_x}, y_{M_x}) inside the macroblock (i,j). The inter-layer predictor is a scaled version of the motion vector $\mathbf{f}_{4,BL}(h,k,M_x)$ computed in the Fine Search for the corresponding block mode M_x of the BL where h and k are the indices of the macroblock of BL with coordinate $[x_{BL}(h,k), y_{BL}(h,k)]$ for the upper-left pixel given by equation (3), while the position of the block inside the macroblock is given by

$$x_{BL}(h,k,M_x) = \frac{x_{EL}(i,j) - x_{CW} + x_{M_x}}{r_X} \qquad y_{BL}(h,k,M_x) = \frac{y_{EL}(i,j) - y_{CW} + y_{M_x}}{r_Y}$$

The value of the predictor is given by:

$$\mathbf{p}_4(i,j,M_x).x = \mathbf{f}_{4,BL}(h,k,M_x).x * r_X \qquad \mathbf{p}_4(i,j,M_x).y = \mathbf{f}_{4,BL}(h,k,M_x).y * r_Y$$

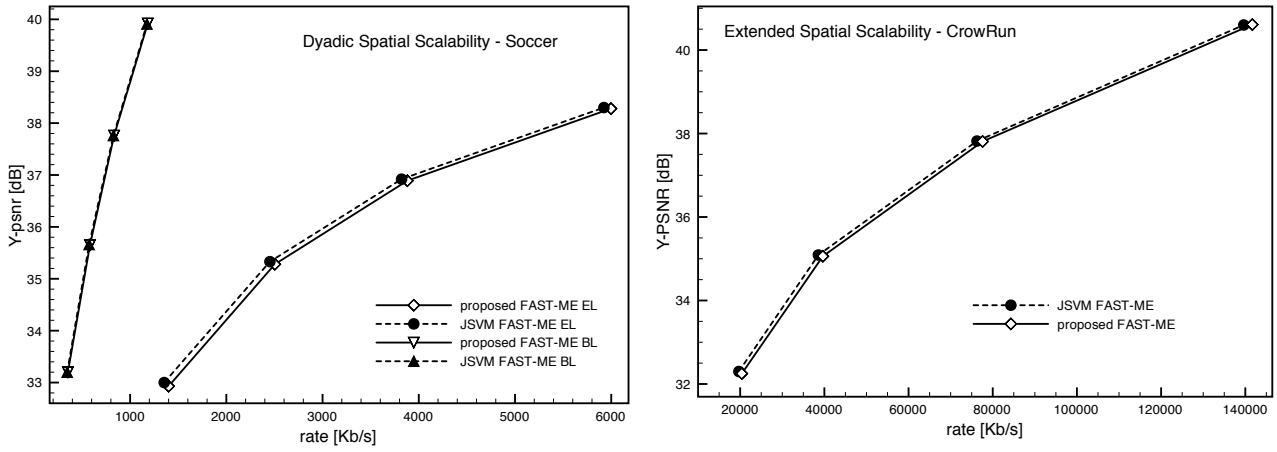
As explained before, if the MB lies outside the cropping window, the motion information of the block is derived using the motion estimation algorithm explained in section 2, with the difference that the Fine Search is performed with temporal predictors set to the temporal predictors of the closest block that lies inside the crop window.

4. EXPERIMENTAL RESULTS

In order to evaluate the performance of the proposed algorithm different configurations have been tested: Coarse Grain Scalability (CGS), Medium Grain Scalability (MGS), dyadic Spatial Scalability (SPA) and Extended Spatial Scalability (ESS), in each case using different test sequences, where the HD sequences used for ESS test are provided by the European Broadcasting Union[7] .

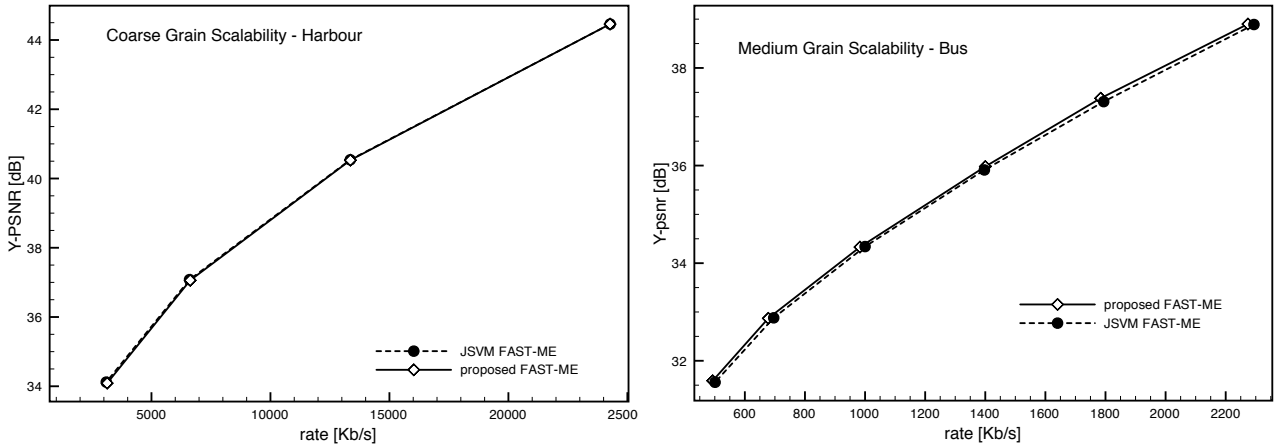
In all the configurations we compare the fast search algorithm used in the JSVM 9.14 reference software[8] and the proposed algorithm in terms of Rate-Distortion (R-D) performance and complexity. More details about how the fast search algorithm adopted in JSVM software works can be found in[5] . The R-D performance is evaluated using the Bjontegaard Delta[9] , as suggested by the JVT committee, while the complexity is evaluated as the number of tested 4x4 block-matches for each macroblock performed during the motion estimation process. It has been decided to evaluate the complexity in terms of number of matches in comparison to the encoding time since currently only a software implementation of the SVC encoder is available, whereas the final target of our work is the hardware implementation for real-time coding. With a software implementation the encoding time strongly depends on the level of code optimization for example efficient implementation of the matching functions. At the moment our algorithms have still to be optimized and so a comparison of the encoding time is not a fair indicator of the complexity reduction. Furthermore, in view of an hardware implementation, the aim is to minimize the number of matches performed for each macroblock because this is the most time-consuming operation involved at the encoder for each macroblock.

The main settings of the JSVM software used for all the configurations are: 4, 8 and 16 picture GOP size with P-type key-pictures, adaptive inter-layer prediction, single loop decoding and intra period usually equal to 2 or 4 times the GOP size. For the SPA e ESS configurations we tested two different encoding modes for the EL: the first one using the same QP for both the BL and EL, while in the second one the QP of the EL is set to the QP of the BL - 6. In the CGS configuration we tested only the case with the QP of the EL equal to the QP of the BL - 6, as suggested in.[8] In MGS configuration we usually define 2 enhancement layers with 3 MGS vectors for each one and the extraction process to obtain the sub-bitstreams has been performed using the "Quality Layers"as suggested in[10] . In terms of resolution and frame-rate, for the SPA test we used a CIF BL at 30Hz and a 4CIF EL at 30Hz or 60Hz; for the CGS and MGS tests both CIF 30Hz and 4CIF 30 Hz while for the ESS test we used a SDTV (720x576) BL at 25Hz and a HDTV EL (1920x1024) at 50 Hz. At the moment, the proposed implementation supports only the progressive mode, and so the SDTV BL used in ESS experiments is not a native PAL/NTSC format, but rather it was obtained by cropping and downsampling the original HDTV video. For the dyadic and Extended Spatial Scalability simulations we used the high complexity version of the



(a) (b)

Figure 5. RD comparison using a GOP size = 8; 5(a): SPA, 5(b): ESS



(a) (b)

Figure 6. RD comparison; 6(a): CGS with GOP size=8, 6(b): MGS with GOP size=16

algorithm, because the different spatial format decreases the inter-layer correlation between the respective motion fields, requiring a more accurate motion estimation in the EL. In CGS and MGS, the low complexity mode of the algorithm has been used because each layer has the same resolution and it is reasonable that the EL needs only a refinement of the BL layer motion information. Therefore fewer motion vectors estimation should provide for enough quality improvement.

Examples of the R-D comparison for SPA, ESS, CGS and MGS configurations are shown in Figures 5(a), 5(b), 6(a) and 6(b), while Table 1 presents the mean number of 4x4 block-match for each MB related to these configurations. As can be noticed in Figures 6(a) and 6(b) the proposed algorithm well adapts to quality scalability configurations (both CGS and MGS), mainly because the frame rate ratio f_R and the resolution ratios (r_X and r_Y) are equal to 1, and the motion information of lower layers is almost the same at the higher layers. For MGS configuration we tested the "forced"inter-layer prediction, that tests only the new inter-layer MB modes at the enhancement layer. Although the complexity gain is lower compared to the other configurations, the reduction of 4x4 match is still considerable. For spatial scalability configurations, we considered a different frame-rate for the BL and the EL. As can be noticed from Figures 5(a) and 5(b) the fast search algorithm in JSVM software works slightly better than the proposed algorithm, especially at low bit-rate. Furthermore, from Figure 5(a) it is interesting to note as the performance of the two algorithms for the base layer are almost the same.

Table 1. Complexity Analysis Examples

Configuration	**4x4 match for MB**		**gain**
	proposed fast ME	**JSVM fast ME**	
Dyatic Spatial scalability	7888	179670	95.6%
Extended Spatial scalability	9645	561685	98.2%
Coarse Grain scalability	7500	337000	97.8%
Medium Grain scalability	10614	81355	87%

Table 2. Summary of performance comparison

Configuration	**Bjontegaard Delta**	
	Rate %	**Y-PSNR**
SPA, GOP 4, QP_EL = QP_BL-6	2,88	-0,11
SPA, GOP 8, QP_EL = QP_BL-6	2,57	-0,10
SPA, GOP 4, QP_EL = QP_BL	3,75	-0,13
SPA, GOP 8, QP_EL = QP_BL	3,08	-0,11
CGS (LC), GOP 4, QP_EL = QP_BL-6	1,59	-0,06
CGS (LC), GOP 8, QP_EL = QP_BL-6	0,72	-0,03
ESS, GOP 4, QP_EL = QP_BL-6	2,77	-0,13
ESS, GOP 8, QP_EL = QP_BL-6	1,86	-0,06
ESS, GOP 4, QP_EL = QP_BL	6,48	-0,30
ESS, GOP 8, QP_EL = QP_BL	1,72	-0,06

All the other performed experiments are summarized in Table 2, where the values have been obtained as the average over different working points, approximately in the 30dB to 40dB range. Table 2 does not present results for MGS scalability since it is a relatively new configuration and the performance comparison is similar to those of CGS scalability. The motion estimation computational gain of the proposed algorithm is not reported in Table 2 because all the tested configuration show an almost constant gain, which is about 96% to 98% complexity reduction with respect to the JSVM Fast-ME method, as also reported in Table 1. Table 2 shows that the proposed algorithm has a good tradeoff between coding efficiency and complexity. In general the performance depends on the motion activity of the sequence, because the higher the motion, the more difficult it is to catch the "true"motion vector by testing few vectors. In the spatial scalabiliy configurations (SPA and ESS) almost all the sequences show a Bjontegaard Delta lower that 4% in bit-rate increasing or $0,15$ dB of Y-PSNR decreasing, while for the CGS configuration the proposed algorithm shows almost the same performance of the Fast Search algorithm in the reference software, and in fact the loss is lower than $0,2dB$ in Y-PSNR or 2% in bit-rate increasing. About the two different modes of the proposed algorithm, the high complexity version increases the number of tested motion vectors by about 50% with respect to the low complexity one, but with a better R-D performance, so that it appears suitable for spatial scalability applications.

5. CONCLUSIONS

This work presents a fully scalable motion estimation algorithm for the Scalable Extension of the H.264/AVC standard. The proposed algorithm correctly works for all the scalability configuration except for progressive to interlaced scalability. Two different modes for the algorithm at the enhancement layer are proposed, the low complexity mode suitable for CGS and MGS and the high complexity mode for spatial scalability (both ESS and SPA).

In conclusion, the proposed algorithm shows good performance with a very high reduction of the complexity and a marginal loss in quality. In particular it has been shown that for CGS and MGS scalability it is possible to obtain the same RD performance, while for spatial scalability configurations the loss in performance is limited within $0,2dB$ in Y-PSNR or 4% in bit-rate increasing. Although at the moment only a software implementation of the encoder is available, the low complexity features shown in the work makes it suitable for hardware implementation in view of its use for consumer applications.

REFERENCES

[1] ITU-T and ISO/IEC JTC 1, "Advanced Video Coding for Generic Audiovisual Services." ITU-T Rec. H.264 and ISO/IEC 14496-10 (MPEG-4 AVC), Version 8 (including SVC extension): Consented in July 2007.

[2] Schwarz, H., Marpe, D., and Wiegand, T., "Overview of the Scalable Video Coding Extension of the H.264/AVC standard," *IEEE Transaction on Circuits and Systems for Video Technology* **17**(9), 1103–1120 (2007).

[3] D.Alfonso, M.Gherardi, A.Vitali, and F.Rovati, "Performance analysis of the Scalable Video Coding standard," *Proc. of 16th Packet Video Workshop* (2007).

[4] L.Lima, D.Alfonso, L.Pezzoni, and R.Leonardi, "New fast search algorithm for H.264 scalable video coding extension," *Proc. Data Compression Conference (DCC)* (2007).

[5] Chen, Z., Zhou, P., and He, Y., "Fast Motion Estimation for JVT." JVT input document G016, March 2003.

[6] Schwarz, H., Marpe, D., and Wiegand, T., "Comparison of MCTF and closed-loop hierarchical B pics." JVT input document P059, July 2005.

[7] European Boadcasting Union. http://www.ebu.ch/en/technical/hdtv/test_sequences.php.

[8] ITU-T, "JSVM 10 software." JVT-W203, April 2007.

[9] Bjontegaard, G., "Calculation of average psnr differences between rd-curves." VCEG contribution M33, April 2001.

[10] Amonou, I., Cammas, N., Kervadec, S., and Pateux, S., "Optimized Rate-Distortion Extraction With Quality Layers in the Scalable Extension of H.264/AVC," *IEEE Transaction on Circuits and Systems for Video Technology.* **17**(9), 1186–1193 (2007).