HDS, a real-time multi-DSP motion estimator for MPEG-4 H.264 AVC high definition video encoding

Fabrice Urban, Jean François Nezan, Mickael Raulet

J Real-Time Image Proc (2009) 4:23–31
DOI 10.1007/s11554-008-0110-0

SPECIAL ISSUE

HDS, a real-time multi-DSP motion estimator for MPEG-4 H.264 AVC high definition video encoding

Fabrice Urban · Jean-François Nezan · Mickaël Raulet

Received: 13 June 2008 / Accepted: 2 December 2008 / Published online: 20 January 2009

Abstract H.264 AVC video compression standard achieves high compression rates at the cost of a high encoder complexity. The encoder performances are greatly linked to the motion estimation operation which requires high computation power and memory bandwidth. High definition context magnifies the difficulty of a real-time implementation. EPZS and HME are two well-known motion estimation algorithms. Both EPZS and HME are implemented in a DSP and their performances are compared in terms of both quality and complexity. Based on these results, a new algorithm called HDS for Hierarchical Diamond Search is proposed. HDS motion estimation is integrated in a AVC encoder to extract timings and resulting video qualities reached. A real-time DSP implementation of H.264 quarter-pixel accuracy motion estimation is proposed for SD and HD video format. Furthermore HDS characteristics make this algorithm well suited for H.264 SVC real-time encoding applications.

Keywords Motion estimation · Digital signal processor · Real-time · H.264 AVC/SVC

F. Urban (✉)
Thomson CR-Video Compression Lab,
1 av. de belle fontaine, CS 17616,
35576 Cesson Sevigne, France
e-mail: fabrice.urban@thomson.net

J.-F. Nezan · M. Raulet
IETR/Image group Lab, UMR CNRS 6164/INSA,
20 av. des Buttes de Coesmes,
35043 Rennes Cedex, France
e-mail: jnezan@insa-rennes.fr

M. Raulet
e-mail: mraulet@insa-rennes.fr

1 Introduction

Eventhough the area of video compression has existed for many decades, programming a coding algorithm is still a challenging problem. With the current communication systems and the improvement of video compression, video broadcasting is more and more widespread. Off-line encoders provide good video quality and compression rates. The actual bottleneck is to provide compressed video in real-time to communication systems. Real time encoders have to cope with timing constraints and HD video formats. Furthermore, the trend of using video everywhere and at any time leads to implement encoders in embedded systems with limited hardware resources. All those constraints have to be solved while keeping a good video quality and compression rates. The investigation and understanding of the foundations of video compression is therefore more important than ever.

In this context, motion estimation (ME) is known to be a key operation. A highly accurate ME can significantly reduce the bit-rate of a video stream, but involves a high computational complexity. The high performance of H.264 is mainly due to improved motion compensation modes such as variable block-size motion compensation, multiple reference pictures and Fractional-accuracy motion estimation (FME) [1]. However the introduction of numerous modes raises the complexity of the codec and makes real-time H.264 compression challenging, especially for high-definition video. On top of that H.264 SVC standard provides scalability features to manage, store and distribute video content towards multiple kinds of terminals and over different access technologies. A single SVC bitstream is used instead of one AVC bitstream per terminal, saving the global available bandwidth. Integer motion estimation (IME) has been widely studied in the past few years. Fast

algorithms have been developed to reduce the computational burden with limited quality loss. The goal of this paper is to study ME algorithms and their use in new standards in terms of both quality and complexity. It is an evolution of previous work [2] in that we introduce new implementation performance results, and a new ME algorithm. An implementation of fast variable block-size is also presented and new application perspectives are proposed.

Video compression has recently become an important feature of 3G cell phones, personal digital assistants, and other battery-powered devices. This kind of devices are very often based on digital-signal processors (DSP) to optimize the performance–consumption ratio. Video codecs are also needed in base stations for inline transcoding or in Real-time H.264 HD video encoding solutions. Here again DSP are widely chosen in multiple components and/or multiple cores hardware platforms. ME algorithms have been prototyped onto DSP TI C6x 1Ghz and results are discussed.

The paper is organized as follows: Sect. 2 is a short state of the art of existing motion estimation techniques, Sect. 3 describes developed embedded implementations and compares motion estimators on 8×8 blocks in terms of quality and complexity. Section 4 gives more results for HDS algorithm with interesting AVC/SVC features like variable block size, subpixel motion vectors and scalability. Finally conclusions and future work are given in Sect. 5.

2 Motion estimation techniques

Motion estimation goal is to find relative motion between two images in order to eliminate temporal redundancy. For video compression where the picture is usually divided into blocks, block matching algorithms (BMA) are most widely preferred. The basic hypothesis are non-deformable objects having an apparent translation in the image plane. One motion vector can then be estimated for each block.

2.1 Block matching

BMA consists in searching for each $M \times N$ block of the current picture a match in a reference picture. A distance measure is computed between the current block and some candidates. This measure is most often the sum of absolute differences (SAD) for its implementation simplicity.

The simplest BMA is the full search where every candidate within a search window of magnitude p pixels is considered. It is very computationally intensive. As the required processing power is too high, a lot of fast algorithms had been proposed using essentially three optimization techniques.

The first one computes a full SAD the least often as possible [3, 4]. It is actually possible to eliminate rapidly bad candidates before the full SAD is computed. These algorithms reduce drastically the number of calculations; however, they use a lot of test operations which make the implementation difficult to optimize and they do not consider memory bandwidth. Thus even if the computation time is statistically improved, the worst case might lead to a worse result than full search, which is unacceptable in a real-time context.

The second one reduces the candidate set by choosing a most likely search direction as soon as possible, under the assumption that the error (SAD) surface is monotonic. As this is not always verified, some algorithms get trapped in local minima. Chen and Al [5] suggest to search in one direction at a time, whereas logarithmic search proposed by Jain and Jain [6] and three step search from Koga and Linuma [7] begin by a coarse estimation then refine the result. In [8] motion is estimated recursively. At each step the SAD for some candidates in a diamond pattern around current position are computed. The motion is recursively refined following the decreasing SAD direction.

The third technique takes video sequence contents into account: motion fields present some continuities (spatially and temporally), so it is possible to predict the movement of a block from the neighboring blocks and previous images. A set of predictors is then available (from the causal neighborhood). Each of them is then evaluated (by calculating the SAD with the current block) and a local search is performed around the best one (which minimizes the SAD) to refine the movement. A lot of algorithms using this technique have been developed [9, 10, 11, 12, 13]. They differ by their predictor sets and their local search patterns. Hierarchical Motion Estimator (HME) [14] introduces a coarse to fine picture definition decomposition to add reliable hierarchical predictors.

Enhanced predictive zonal search (EPZS) [10] and HME are particularly interesting for a DSP implementation. They are the two finally selected techniques from which a third one is derived. They are more precisely detailed in the following.

2.2 EPZS

Enhanced predictive zonal search algorithm is an improvement of PMVFAST [11] algorithm thanks to new predictors. The prediction step is consequently more accurate and the local search (Fig. 1 left) is thus reduced. Coarse refinement with a large diamond pattern in PMVFAST is unnecessary. The best predictor is directly refined using a small diamond or square pattern (Fig. 1 right). The improvement of prediction step reduces execution time. The refinement step consists in a recursive diamond search:

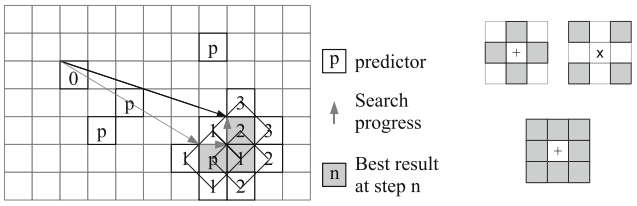


Fig. 1 EPZS principles

the current best vector (initialized with the prediction step) is compared to its neighbors according to the test pattern and the best vector becomes the new search center.

An early stopping criterion already present in PMV-FAST speeds-up the operation by avoiding unnecessary computations. The process is stopped as soon as the result is "good enough", i.e. SAD is lower than an adaptive threshold. The execution time is thus low but highly depends on the video sequence. This motion estimator has a low computation load and thus is a good candidate for a fast software implementation of motion estimation. EPZS is used in software video encoders such as XviD and JM H.264 reference software.

In our EPZS implementation, the early stop criterion has not been implemented in order to get a constant execution time. As a consequence the quality is slightly increased.

2.3 HME

The hierarchical motion estimator (HME) [14] is based on a multi-level refinement process where the motion vectors are first coarsely estimated on a sub-sampled picture. The algorithm starts by building a pyramid of pictures (Fig. 2). Level 0 is the full-resolution picture, the level $n + 1$ is level n low-pass filtered and sub-sampled picture.

A sub-sampled motion field is firstly estimated on the low resolution picture (highest level), then the motion field is successively refined. The block size through the pyramid is constant so that global motions are detected on the coarsest levels and refinement is achieved when resolution is increased. At each resolution level, a predictive motion

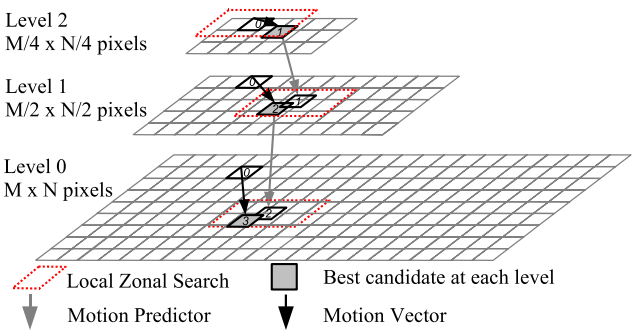


Fig. 2 Pyramid of pictures in HME

estimation is performed, as for EPZS, with the difference that reliable hierarchical predictors are provided from lower resolution level.

The refinement step is a reduced full search around the best predictor. In addition to the motion estimation operations, HME implementation takes into account the computation of the sub-sampled pictures pyramid. Each level is a sub-sampled picture of the lower level's to which a 3-tap Gaussian low-pass filter is applied.

The local search and predictive mechanism in EPZS and HME naturally provide a low entropy homogeneous motion field. This is an advantage for video compression.

2.4 HDS algorithm

The two methods precedently presented have each their advantages. EPZS is very fast thanks to its recursive diamond search window but large motion vectors can not be found due to its limited search window. HME search window is not limited but its local full search requires more processing power. To combine the interest of both and to keep their advantages, we propose here a new algorithm based on a HME and EPZS combination: Hierarchical Diamond Search (HDS) is a recursive diamond search applied to a multi-level decomposition.

The multi-level decomposition provides robust prediction. The reduced resolution levels on the top of the pyramid (Fig. 2) allow the detection of large motion with only a reduced search window. In the coarsest resolution level, a reduced full search search is performed to catch very large motion. Because the image size is reduced, the impact on computation cost is negligible. As an example, in high definition, a four-level pyramid with a search range of ± 16 in the coarsest level can catch small objects with motion as large as 128 pixels of amplitude. The multi-resolution approach provides robust hierarchical predictors.

At each level the motion is estimated block per block, and for each block, the motion is first predicted from hierarchical and spatial predictors. The first kind of predictors provides large motion detection abilities whereas the second one provides accurate information on the neighborhood movements and favors a homogeneous motion field and fast convergence of the algorithm. The different predictors are evaluated on a SAD basis and the best one is selected for a refinement step.

The refinement step consists in recursively trying a small displacement around the current best motion vector. This local search is initialized with the results of the predictive step. Then, at each iteration, a displacement of one pixel in every direction (eight neighbors) is analyzed. The best position is chosen as the new search center of the recursive process. If the best position is already at the center of the pattern, the search stops. To ensure limited

calculation, the number of iterations can be bounded to a maximum value.

The hierarchical approach together with fast diamond search ensure both a robust predictive step and a fast processing. The resulting motion field is reliable and close to the physical motion.

3 Real-time implementation on DSP

Real-time motion estimation implementation for MPEG-4 H.264 AVC [15] high definition video encoding is challenging. With tools such as variable block size, quarter-sample accuracy and multiple reference pictures, motion estimation needs high computation power and memory bandwidth. The detailed implementations of the three algorithms are given in Table 1. Several SD (576p: 720 × 576) and HD (720p: 1,280 × 720) video sequences have been used to test HDS performances and compare them to HME and EPZS ones. The content of sequences varies from high and complex movement (Formula1, Football) to small motion (RaidMaroc, Horses).

Results are presented with two distinct criteria: execution time and motion estimation quality. Quality is more important in high end solutions such as video broadcasting whereas for low cost solutions, execution time (or algorithm complexity) must be kept low. In order to provide consistent results, motion estimators have been implemented and optimized onto a Texas Instrument TMSC6416 DSP at 1 Ghz. For quality comparisons, the motion estimators have been implemented in the JM H.264 video encoder. In this section only 8 × 8 inter-frame coding mode is allowed on P frames to eliminate the influence of a decision algorithm and intensively stress the motion estimator. Others tools will be discussed later on.

3.1 DSP optimizations

HME and EPZS have been implemented and optimized for TI C64x DSP. Loops have been optimized using SIMD vectorization and loop unrolling. Compilation process have been optimized with specific key-words like "#pragma", "restrict", "inline" and "const", and memory accesses have been enhanced using cache, on-chip memory and Enhanced direct memory access peripheral (EDMA) transfers. Execution times have been reduced by a factor of five compared to the original code with only straightforward compilation optimizations. Results will be detailed in the following section.

Table 1 Implementation details

Zonal search	Predictors
EPZS: 1 resolution level	
8-neighbors pattern Diamond search	1 temporal
No early stop	4 spatial
HME: 4 resolution levels	
Full search ±16 at lowest resolution	5 hierarchical
Reduced full search (±3)	4 spatial
HDS: 4 resolution levels	
Full search ±16 at lowest resolution	5 hierarchical
8-neighbors pattern Diamond search	4 spatial

To compute the multi-resolution image pyramid, a separable filter has been implemented. The chosen 3-tap low-pass filter is optimized together with the sub-sampling in order to compute only needed samples (every other pixel horizontally and vertically) and reduce constraints on memory bandwidth. Furthermore, memory access is also optimized for high definition resolution where data is located in external memory. In this case, computations are performed concurrently with memory accesses using the on-chip EDMA.

H.264 FME feature allows the use of quarter-sample precision motion vectors. The compression efficiency is highly improved at the cost of a higher computational complexity. The picture definition is increased by interpolating successively at half-sample accuracy with a 6-tap filter and at quarter-pel accuracy with a linear filter. The quarter-pel precision can be achieved by different means; the first one is to search directly in the interpolated picture with a search window four times as large horizontally and vertically. The second one consists in two steps [16]: the motion estimation is firstly performed at pixel accuracy then refined at sub-pixel accuracy. This last method allows to compute sub-pixel samples "on the fly" meaning that half and quarter pixel positions are interpolated only when needed. As a consequence computations are slightly increased whereas memory bandwidth is drastically reduced which is a good trade-off for a DSP implementation. To reduce further execution times, the sub-pixel interpolation filters may be replaced by linear filtering which reduces computations while slightly reducing interpolation accuracy. This latter point is not discussed further here.

3.2 Execution times

Figure 3 gives the execution times for three implemented motion estimators on SD (720 × 576) and HD (1,280 × 720) progressive image sequences. For each algorithm, pixel and quarter-pixel accuracy versions have been considered. Motion Estimation is performed onto 8 × 8 blocks. Regardless the resolution and accuracy, EPZS Motion estimator is faster than HME and HDS. It can be partly explained by the computation of the multi-resolution pyramid and the motion estimation of lower levels. In addition, HME also has more predictors and a zonal search inducing more computations.

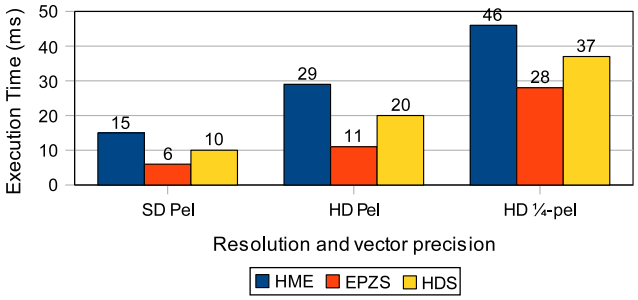


Fig. 3 Execution time comparison of the motion estimators

EPZS and HDS implementations at quarter-sample accuracy reach respectively more than 30 and 25 frames per second on a DSP for high definition video. For a HME implementation at 30 frames per seconds, the processing power of at least two DSPs is needed, for example in a two-stage pipeline composed of hierarchical levels in the first stage and one full resolution level in the second. The full search executed at a given level in HME leads to more calculations than HDS Diamond Search. Both HME and HDS hierarchical algorithms are more complex than EPZS.

3.3 Motion estimation quality

In order to evaluate the quality of motion vector fields for each technique, each motion estimator has been integrated in an H.264 encoding software. For the comparison purpose, P and I frame are allowed, with one I frame every 25 frames. In P frames only 8 × 8 inter-frame mode is activated to stress the motion estimator and compare only motion field quality, with no decision interfering. Rate control has also been deactivated so that rate/distortion curves based on PSNR reflect the ability of the motion estimator to find a good match.

A few sequences have been encoded to evaluate the compression performances of h.264 encoding with the motion estimators. SD formula1 and HD football are high motion sequences with traveling and many moving objects. These sequences highlight the matching ability of motion estimators. SD RaidMaroc and HD horses are slow motion sequences with few moving objects in favor of low entropy motion fields. All these sequences are common content and must be well handled by the encoder. Figure 4 shows the rate/distortion curves corresponding to the first 200 pictures of two sequences: SD fomula1 and HD football. For each sequence, the quality (mean PSNR) is plotted against the mean bit-rate. Figure 5 sums up results for several SD and HD video sequences. It represents the data-rate increase reached at a given quality (constant PSNR). It is expressed using the difference with the data-rate reached with the encoding software based on HME motion estimation solution.

These results show that EPZS algorithm lead to a bit-rate increase of almost 20% for high motion sequences, which is unacceptable for high-end solutions. With results comparable to HME (less than 4% bit-rate increase on the worst case and 1.5% decrease on the best case), HDS appears to be a good trade-off between encoding performances and processing time. This illustrates the ability of HDS to find an appropriate vector in case of high motion sequences and a low entropy motion field in case of slow motion. Hierarchical algorithms have a more robust prediction step whereas EPZS has a limited search range and relies on temporal prediction. Therefore, HME and HDS perform a lot better than EPZS at scene cuts.

The reduced range of the motion vectors well handled by EPZS is a limitation in high motion sequences. Moreover, the displacement between a frame to encode and a reference frame will be increased when using B pictures, and even more in case of hierarchical GOP structure because the motion vector amplitude increases. Therefore, EPZS-based techniques need to be associated to a vector-tracing technique [17] to reach efficient vector prediction. It may involve the estimation of motion fields not used by the video encoder but by the motion estimator prediction step only, thus increasing processing load.

Motion estimation performances of HME and HDS are comparable, and outperform EPZS algorithm. Performances improvement of HDS over EPZS is worth its slight increase in computational complexity. HDS motion estimation algorithm is therefore chosen for the rest of this paper.

4 MPEG-4 H.264 motion estimation features

On one hand, variable block-size and quarter-pixel motion compensation bring effective compression gain among the

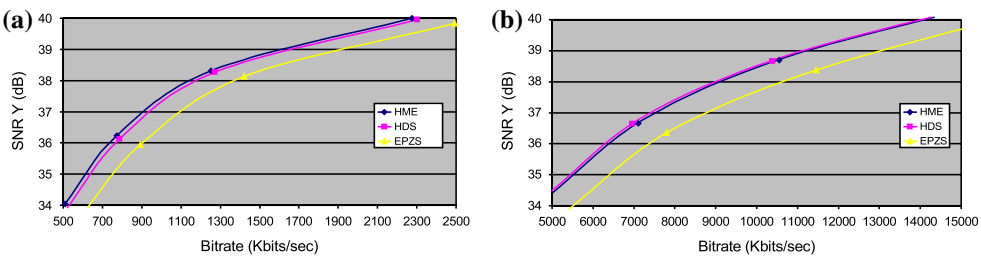


Fig. 4 Rate/distortion curves for SD and HD sequences

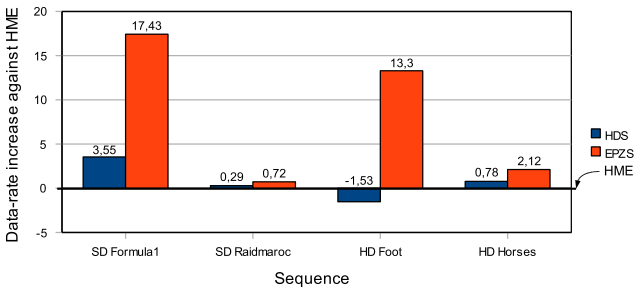


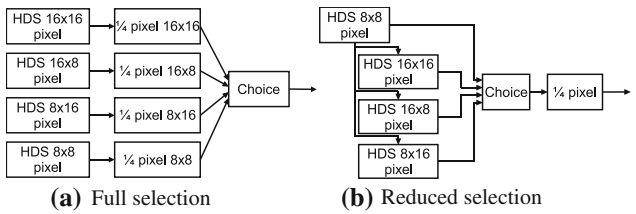
Fig. 5 Encoding results for SD (576p) and HD (720p) video

various coding tools of AVC [18]. On the other hand, the computational complexity of the motion estimation operation is consequently highly increased, making real-time implementation of this operation challenging for high definition video. Scalable extension of H.264 encoding (SVC) allows partial transmission of a video stream thanks to temporal, spatial or quality scalability. Spatial scalability increases motion estimation constraints with the need to compute motion fields at different resolution levels. This section discusses algorithm optimizations of these two specificities.

4.1 Variable block-size and quarter-pixel

Despite the need for complex interpolation operations, the quarter-pixel vector refinement is required in a high-end video coder. Variable block-size, however, magnifies the implementation constraints. For motion compensation in the H.264 standard, 16×16 pixel macro-blocks can be divided in 16×8, 8×16 or 8×8 partitions. The last one can be further split in 8×4, 4×8 or 4×4 blocks. Choosing small blocks improves motion compensation but increases the coding cost as more motion vectors need to be transmitted. For high definition video, it has been shown that block-size smaller then 8×8 brings little compression improvement considering the complexity it brings. Consequently, this section describes a variable motion estimator handling 16×16 to 8×8 block sizes.

The straightforward implementation of the variable block-size motion estimator leads to the solution presented in Fig. 6a: the scheme for one block-size is repeated for every block-size. The main drawback is the increased amount of computation and memory bandwidth.



(a) Full selection **(b)** Reduced selection

Fig. 6 Variable block-size implementation

The first optimization is to start motion estimation including multi-resolution levels with one block-size at integer-pixel accuracy. The results serve then as accurate prediction for other block sizes and hierarchical level motion fields are reused without re-computation.A good initial block-size must be not too big to catch small objects motion and not too small to be less sensitive to noise. Moreover a block size closer to the deduced other sizes (4×4 to 16×16) statistically improves prediction accuracy. Therefore, we chose 8×8 size as a first motion estimation step.

Second, fractional-pixel refinement implies heavy interpolation operations and matching evaluation. In variable block-size, every output vector must be refined to quarter pixel accuracy to avoid compression performances drop. To reduce computation complexity, we choose to select the best motion compensation partition after pixel-accuracy results in order to refine only one block-size per macro-block and thus avoid unnecessary calculation (Fig. 6b). Therefore it drastically reduces computation burden with limited impact if any on compression performances.

Figure 7 shows the typical result of this computational reduction on compression performances. Variable block-size is activated on the encoder. For the reduced selection variable block-size algorithm, the block size is chosen at the motion estimator, otherwise we let the encoder decide. The rate/distortion curve shows clearly the improvement of quarter-pixel refinement for variable block-size on compression performances. The impact of the reduced selection algorithm appears to be very limited and negligible.

Figure 8 is a comparison of execution times on the TI DSP. Thanks to the reduced selection quarter-pixel refinement, the motion estimation process is accelerated by a factor of two compared to the full selection implementation without modifying the hardware. The overhead of sub-pixel refinement is thus minimized compared to the full sub-pixel refinement.

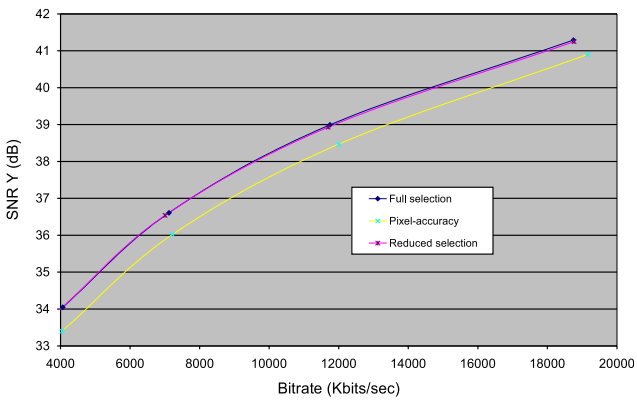


Fig. 7 HDS variable block size mode on HD sequence Football

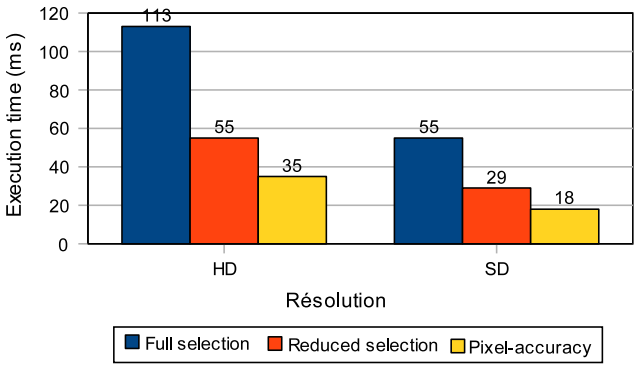


Fig. 8 HDS Variable block-size implementation comparison

The reduced selection solution drastically lowers the computational complexity of variable block-size and quarter pixel motion estimation for video compression with very low impact on compression performances if any. For one reference frame, this solution reaches 30 frames per second on one DSP in standard definition. For high definition, a solution involving two DSPs reaches real time for 720p at 30 fps.

4.2 Scalability

In this article motion estimation is studied for H.264 AVC compression. Nevertheless, the performances of ME for SVC may be deduced from these results. The SVC extension is built on H.264 AVC and re-uses most of its innovative components. As a distinctive feature, SVC generates an H.264 AVC compliant base layer and one or several enhancement layer(s). The base layer bitstream corresponds to a minimum quality, frame rate, and resolution, and the enhancement layer bitstreams represent the same video at gradually increased quality (fidelity scalability) and/or increased resolution (spatial scalability) and/or increased frame rate (temporal scalability) (Fig. 9). Several layer decompositions have been studied in scalim@ges[1] project to optimize the quality reached for each SVC layer. This french project has elaborated scenarii in [19]. In this project, the one selected for digital television is made of three SVC layers. Layer 3 handles a HD 720p at 6 Mbits/s. The video is downsampled by two horizontally and vertically (dyadic decomposition) to reach a 640 × 360 resolution at 1.5 Mbits/s for the SVC layer 2 and a 320 × 180 resolution at 500 kbits/s for H.264 AVC compatible base layer (Layer 1). ME has to be done on each SVC layer. With EPZS for example, ME is done on each layer independently. So each layer will require one or several reference frames where ME is computed. This solution is the most memory and time consuming.

[1] Scalim@ges is a project from the "Media and Network" Cluster in France.

The dyadic decomposition corresponds to the pyramid computed in HME and HDS. To optimize both motion estimation and compression operations, we propose to use a unified multi-resolution decomposition for these two operations with an appropriate motion estimator. HDS algorithm provides hierarchical predictors that can be used in SVC layers. Consequently, there is no more computation time overhead for HDS compared to EPZS because motion estimation has to be done for each SVC layer. In this context HDS provides better quality and saves computation resources. Hierarchical prediction in the motion estimation process favors lower motion vector cost for spatial enhancement layers of SVC.

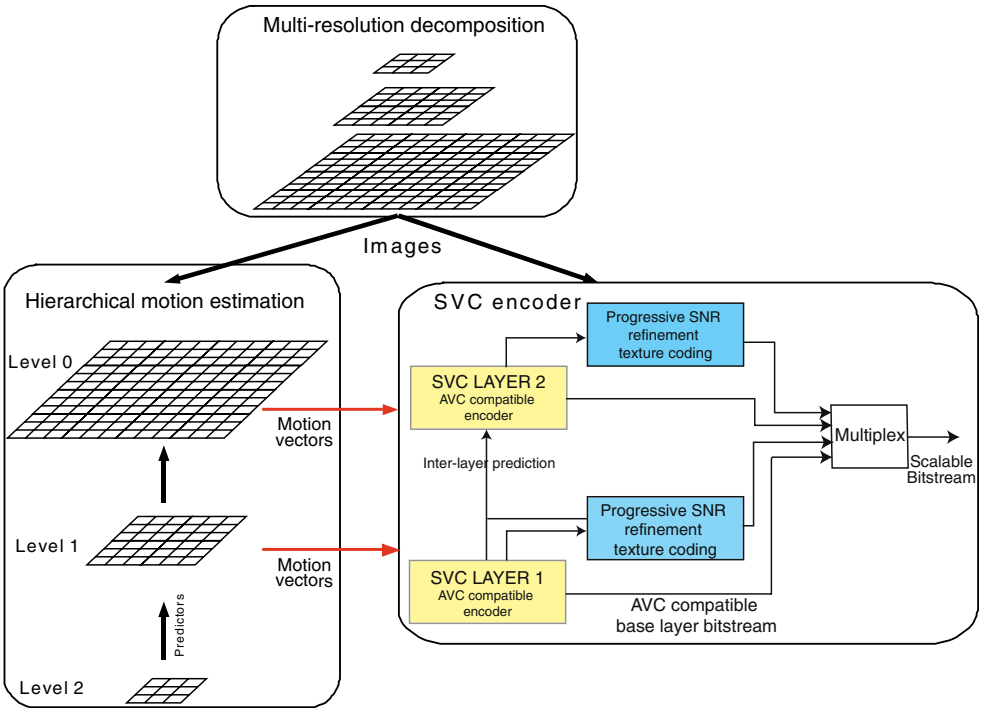
A multi-resolution decomposition is computed once for both motion estimation and compression. It uses either the low-pass filter used in the SVC reference software [20], well suited for non-dyadic mode but very complex for a real-time implementation, or the one implemented in HDS to save computation time. The motion estimator can then be tightly coupled to the encoding loop to use actually encoded vectors as spatial and hierarchical predictors. The motion vector costs are consequently reduced. Using the proposed unified decomposition reduces computation complexity, and in the same time it should improve compression performances.

5 Conclusion

A state of the art of motion estimation techniques has been drawn. HME and EPZS techniques present good quality and computational complexity. They have been prototyped on a 1 Ghz TI C64x DSP and integrated in an H.264 video encoder. Their performances concerning both their execution time and result quality have been compared. EPZS can reach 30 frames per second for HD definition at quarter-pixel accuracy an up to 77 frames per second at pel accuracy. It is a good candidate for a low cost video encoder. HME needs more computational power, but is a good candidate for a high-end video encoder. Compression gain brought by quarter-pel accuracy is worth its computationally expensive implementation. HME regularity would make its implementation interesting onto highly parallel hardware implementations (FPGA, ASIC) whereas HDS appears to be more interesting for software implementations. HDS is a good compromise between motion estimation quality and computation complexity.

For variable block size motion estimation the scheme used for 8 × 8 blocks can be duplicated for each block size. The overhead due to the hierarchical levels processing of HDS is then reduced because it can be done only once for 8 × 8 block size and serve as hierarchical predictors for all the other block size searches.

Fig. 9 SVC global view



The results presented in this paper shows that Motion Estimation for a H.264 coder using full features can be realized in real time for SD video. A 30 frames per second quarter-sample precision motion estimator for HD H.264 video encoding can be prototyped onto two DSP using HDS. As a result, a complete HD real time coder can now be mapped onto a multicore DSP like TI TMS320TCI6487. Another future work will be the study of HDS in a H264 SVC context.

References

1. Richardson, I.E.G.: H.264 and MPEG-4 video compression: video coding for next-generation multimedia. Wiley, New York (2003)
2. Urban, F., Poullaouec, R., Nezan, J.F., Déforges, O.: Real-time multi-dsp motion estimator for mpeg-4 avc/h.264 high definition video. In: International Conference on Signals and Electronic Systems, (2006)
3. Li, W., Salari E.: Successive elimination algorithm for motion estimation. IEEE Trans. Image Process. **4**, 107–110 (1995)
4. Chen, Y.-S., Hung, Y.-P., Fuh, C.-S.: Fast block matching algorithm based on the winner-update strategy. IEEE Trans. Image Process. **10**, 1212–1222 (2001)
5. Chen, M.J., Chen, L.G., Chiueh, T.D.: One-dimensional full search motion estimation algorithm for video coding. IEEE Trans. Circuits Syst. Video Technol. **4**, 504–509 (1994)
6. Jain, J.R., Jain, A.K.: Displacement measurement and its application in interframe coding. IEEE Trans. Commun. **COM-29(12)**, 1799–1808 (1981)
7. Koga, T., Linuma, K., Hirano, A., Iijima, Y., Ishiguro, T.: Motion compensated interframe coding for video conferencing. In: Proceedings of National Telecommunication Conference, vol. NTC81, pp. G5.3.1–G5.3.5 (1981)
8. Tham, J., Ranganath, S., Ranganath, M., Kassim, A.: A novel unrestricted center-biased diamond search algorithm for block motion estimation. IEEE Trans Circ Syst video Technol **8**(4), 369–377 (1998)
9. Hosur, P., Ma, K.: Motion vector field adaptive fast motion estimation. In: Second International Conference on Information, Communications and Signal Processing (ICICS'99) (1999)
10. Tourapis, A.M.: Enhanced predictive zonal search for single and multiple frame motion estimation. In: Proceedings of Visual Communications and Image Processing. pp. 1069–1079 (2002)
11. Tourapis, A.M., Au, O.C., Liou, M.L.: Predictive motion vector field adaptive search technique (PMVFAST). In: Proceedings of Visual Communications and Image Processing (VCIP'01) (2001)
12. Chen, Z., Zhou, P., He, Y.: Fast motion estimation for JVT. JVT-G016.doc (2003)
13. Virk, K., Khan, N., Masud, S., Nasim, F., Idris, S.: Low complexity recursive search based motion estimation algorithm for video coding applications. In: Proceedings of 13th European Signal Processing Conference, Antalya, Turkey (2005)
14. Chupeau, B., Robert, P., Pecot, M., Guillotel, P.: Multiscale motion estimation. In: Workshop on Advanced Matching in Vision and Artificial Intelligence, Munich, 5th, 6th June (1990)
15. Joint Video Team of ITU-T and ISO/IEC 14496-10 "Draft of version 4 of H.264/AVC" Tech. Rep., Nov (2004)
16. Choi, W.I.L., Jeon, B., Jeong, J.: Fast motion estimation with modified diamond search for variable motion block sizes. In: International Conference on Image Processing, vol. 2, pp. 371–374 (2003)
17. Mattavelli, M., Zoia, G.: Vector-tracing algorithms for motion estimation in large search windows. IEEE Trans Circuit Syst. Video Technol. **10**(8), 1426–1437 (2000)

18. Sullivan, G., Wiegand, T.: Video compression—from concepts to the H.264/AVC standard. Proc. IEEE **93**, 18–31 (2005)
19. ISO/IEC JTC1/SC29/WG11: "Svc verification test report," MPEG, Antalya, Tech. Rep. N9577, January (2007)
20. ISO/IEC JTC1/SC29/WG11: "Mpeg-4 video verification models version 18.0," MPEG, Pisa, Tech. Rep. N3908, January (2001)

Author Biographies

Fabrice Urban is a research engineer at Thomson Corporate Research in Rennes (France). He received his postgraduate certificate in signal, telecommunications, images, and radar sciences from Rennes University in 2004, and his Engineering degree in electronic and computer engineering from INSA, Rennes Scientific and Technical University in 2004. He received his Ph.D. degree in electronics and industrial informatics in 2007 from the INSA. His research interests include implementation and prototyping of motion estimation algorithms on multi-component platform.

Jean-François Nezan is an Assistant Professor at National Institute of Applied Sciences of Rennes (INSA) and a member of the IETR laboratory in Rennes. He received his postgraduate certificate in signal, telecommunications, images, and radar sciences from Rennes University in 1999, and his engineering degree in electronic and computer engineering from INSA-Rennes Scientific and Technical University in 1999. He received his Ph.D. degree in electronics in 2002 from the INSA. His main research interests include image compression algorithms and multi-DSP rapid prototyping. He is involved in the ISO/IEC JTC1/SC29/WG11 standardization activities (better known as MPEG) especially in the Reconfigurable Video Coding (RVC) working group.

Mickaël Raulet received his postgraduate certificate in signal, telecommunications, images, and radar sciences from Rennes University in 2002, and his Engineering degree in electronic and computer engineering from National Institute of Applied Sciences (INSA), Rennes Scientific and Technical University. In 2006, he received a Ph.D. degree from INSA in electronics and signal processing in collaboration with the software radio team of Mitsubishi Electric ITE (Rennes, France). He is currently in the Institute of Electronics and Telecommunications of Rennes (IETR) where he is a research engineer in rapid prototyping of standard video compression on embedded architectures (multi DSP architectures). Since 2007, he is involved in the ISO/IEC JTC1/SC29/WG11 standardization activities (better known as MPEG) as a Reconfigurable Video Coding Expert. His interests include video standard compression and telecommunication algorithms and rapid prototyping on multi-DSP architectures from Texas Instruments.