Computational-Complexity Scalable Motion Estimation for Mobile MPEG Encoding

Stephan Mietens[1], Peter H. N. de With[2], Senior Member, IEEE, and Christian Hentschel[3], Member, IEEE

Abstract — *Complexity scalability algorithms are important for mobile consumer devices using MPEG video coding, because they offer a trade-off between picture quality and the embedded available computational performance. This paper presents a new scalable three-stage motion estimation technique, which includes preprocessing of frames in display order and approximation of motion-vector fields using multiple temporal references. A quality refinement of the approximation is added as an optional stage. Furthermore, we present a new scalable motion-estimation algorithm, based on simple edge detection, for integration into the above-mentioned new three-stage motion estimation technique. The complete system provides a flexible framework with a large scalability range in computational effort, resulting in an output quality that scales up smoothly with the number of operations spent on the motion estimation process. Experiments show a scalable computational effort from below one SAD (sum of absolute difference) computation per macroblock up to 15 SAD computations, resulting in a global variation of 7.4 dB PSNR in picture quality (with the "Stefan" sequence). In high-quality operation, the new algorithm is comparable to (or even outperforms) a full-search motion estimation with a search window of 32x32 pixels.*

Index Terms — Complexity scalable, motion estimation, MPEG coding, mobile.

I. Introduction

New example video applications such as internet-based video conferencing, portable television applications and mobile consumer terminals all have different picture quality requirements. These differences can be combined in a single MPEG encoder design by scaling the algorithmic complexity of the applications. Depending on the application, a certain quality loss can be acceptable under circumstances as indicated in the following examples. Let us consider the design of a programmable multi-window TV system. A part of the available general-purpose computation power of a (portable) TV can be saved by reducing the computational effort of the main task (video window), in order to perform smaller secondary tasks (like MPEG encoding, surveillance application) in parallel. A second example that could benefit from complexity scalability is a mobile device with a small display, which receives an MPEG stream having a higher resolution than the display. In this case, still a full and thus costly processing of the video is performed, although it is not needed because the viewer cannot perceive the fine video details of the stream.

It is our objective to design a scalable MPEG encoding system, featuring scalable video quality and a corresponding scalable resource usage [1], [2]. Such a system enables advanced video encoding applications on a plurality of low-cost or mobile consumer terminals, having limited resources (available memory, computing power, stand-by time, etc.) as compared to high-end computer systems or high-end consumer devices. Note that the advantage of scalable systems is that they are designed once for a whole product family instead of a single product.

A computationally expensive corner stone of an MPEG encoder is motion estimation (ME), which is used to achieve high compression ratios by employing the computed motion vectors (MVs) in a recursive temporal prediction loop. We consider ME as a signal-processing task applying three issues as follows, where each issue can contribute to enhance ME with scalability.

- Accuracy level: the accuracy of (candidate) MVs are evaluated in order to decide whether the vectors are suitable to describe picture block displacements or not.
- Vector-selection level: an algorithm provides the rules which MVs are evaluated, in order to estimate the motion between two frames.
- Structural level: the structure of the group-of-pictures (GOP) defines which MV fields (MVF) are needed for the MPEG encoding process. The computation of according MVFs is initiated at this level.

In the past, many new ME techniques have been published for enhancing the vector selection (e.g. [3]-[6]), which aim at reducing the number of MV evaluations that are required for high quality ME. Besides special block-matching criteria such as [7], computational complexity scalability has been introduced for the ME process by techniques that provide a scalable matching criterion on the vector-selection level and/or the accuracy level [8], [9]. In [10], a structural-level ME technique is presented, which processes MVFs twice for quality improvements.

[1] Stephan Mietens is with Philips Research Labs. Eindhoven, NL-5656 AA Eindhoven, The Netherlands.
[2] Peter H. N. de With is with Eindhoven University of Technology/ LogicaCMG, NL-5600 MB Eindhoven, The Netherlands.
[3] Christian Hentschel is with Cottbus University of Technology, D-03044 Cottbus, Germany.

Original manuscript received June 13, 2003
Revised manuscript received November 13, 2003

In this paper, we present two new techniques to insert complexity scalability into the ME process. The first technique is located on the structural level and processes MVFs in three stages. The first stage performs initial ME with the input video frames in display order ("IBBP") and independent of GOP (group of pictures) structures. The second stage efficiently derives MPEG MVFs by multi-temporal approximations, based on the MVFs that are computed in the first stage. Furthermore, the quality of full-search ME can be obtained with an optional third refinement stage. The aforementioned stages form our new Scalable MVF Approximation and Refinement Technique (SMART).

The second ME technique is located on the vector-selection level and provides Content-Adaptive REcursive Scalable ME (CARES) through block classification based on edge detection. Prior to estimating the motion between two frames, the macroblocks inside a frame are classified into areas having horizontal, vertical edges or no edges. The classification is exploited to minimize the number of motion vector evaluations for each macroblock by e.g. concentrating vector evaluations across the detected edge. A novelty in the algorithm is a *distribution* of good motion vectors *to* other macroblocks, even already processed ones, that differs from other known recursive ME techniques that reuse MVs *from previously* processed blocks.

The paper is organized as follows. Section II gives a brief introduction to the ME process in MPEG encoders and Section III addresses the problem statement. Section IV presents the new SMART technique to process motion estimation with considerable savings in computational effort and memory bandwidth for resource-constrained applications. A block classification based on edge detection to support the ME is presented in Section V. The new CARES algorithm that makes use of the block classification is presented in Section VI. Section VII shows experimental results and Section VIII concludes the paper.

II. MOTION ESTIMATION FOR MPEG

Figure 1 shows the basic architecture of an MPEG encoder. The ME process in MPEG systems computes translational displacements of macroblocks between frames, which are expressed as motion vectors. For each macroblock, a number of candidate motion vectors are examined to find a best match, usually based on the Sum of Absolute Differences (SAD) of two macroblocks. The array of motion vectors for all macroblocks of a frame forms a Motion Vector Field (MVF).

The MPEG coding standard defines three different types of frames, namely I-, P- and B- frames, where I-frames are coded without any temporal reference (completely independent). P-frames are based on ME using one temporal reference, namely the previous reference frame. B-frames can refer to both the previous and the upcoming reference frame. Reference frames are I- and P-frames. Since B-frames refer to future reference frames, they cannot be (en/de)coded before this reference frame is received by the (en/de)coder. Therefore, the video frames are processed in a reordered way, e.g. "IPBB" (transmit order) instead of "IBBP" (display order).

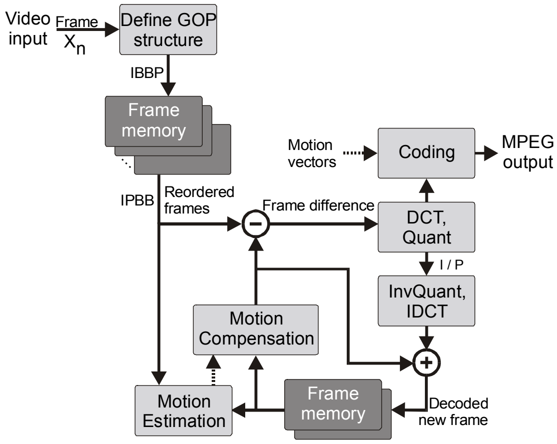

Fig. 1. Basic architecture of an MPEG encoder.

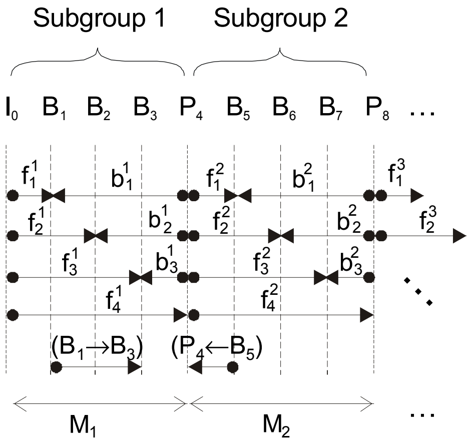
Subgroup 1
Subgroup 2
I_0 B_1 B_2 B_3 P_4 B_5 B_6 B_7 P_8 ...
f_1^1 b_1^1 f_1^2 b_1^2 f_1^3
f_2^1 b_2^1 f_2^2 b_2^2 f_2^3
f_3^1 b_3^1 f_3^2 b_3^2
f_4^1 f_4^2
$(B_1 \rightarrow B_3)$ $(P_4 \leftarrow B_5)$
M_1 M_2 ...

Fig. 2. Example of vector fields used for motion estimation in MPEG encoding after defining a GOP structure. In this example, a GOP with a constant $M = 4$ was chosen.

For ease of discussion, we form Sub-Groups-Of-Pictures (SGOP) that have the form $(I/P)BB...B(I/P)$ within an MPEG GOP. For the same reason, we address pictures as frames, although interlaced pictures have two fields. For further explanation, we refer to Figure 2. The prediction depth of a subgroup k is denoted by M_k, analogous to the prediction depth M of a GOP, and can vary from SGOP to SGOP. The MPEG forward vector-field, which is used in the prediction of the i^{th} frame, is denoted by $\underline{f_i^k}$. The MPEG backward vector-field is denoted by $\underline{b_i^k}$. Arbitrary MVFs are denoted by $(X_m \rightarrow X_n)$ for the forward case and $(X_m \leftarrow X_n)$ for the backward case, indicating motion between frame X_m and X_n with $n > m$. If the frame type is known, the arbitrary frame type X can be replaced by I, P, or B. All given indices (k, i, m, n) may left out if they are not needed.

III. Problem statement

A large number of algorithms has been proposed for reducing the computational effort of a full-search ME. The algorithms make a trade-off between complexity and the quality of the computed vector fields. When compared to full search, popular algorithms like New Three Step Search [3] and Center-Biased Diamond Search [4] provide a good quality of motion vector fields at low cost. However, the accuracy of the MVs is limited for fast motion in the video sequence.

A further reduction of the computational effort has been achieved by using Recursive ME (RME, already discussed in [6], [5], [11], and now included in MPEG-4), that derives candidate MVs from previously computed MVs in both the current MVF ("spatial" candidates) or the previous MVF ("temporal" candidates). Up to now, the RME algorithms have been used on GOP structures with fixed M and B-frames were not considered for long-term tracking of the motion. Therefore, only $(I/P \rightarrow I/P)$ vector fields $\underline{f}_M^k$ have been used for the computation of the next SGOP vector fields $\underline{f}_M^{k+1}$, because they have the same temporal distance.

A more sophisticated approach for ME is presented in [10], featuring a two-step estimation process and enhanced vector-field prediction. The first step of this approach is a coarse RME to pre-estimate the forward vector-fields within an SGOP. The second step uses the vector fields computed in the first step as prediction and performs an additional RME. Vector fields that are used as prediction are scaled to the appropriate temporal distance that is actually needed.

The problem of the aforementioned ME algorithms is that a higher value of M increases the prediction depth, implying a larger frame distance between reference frames, thereby hampering accurate ME. Furthermore, the algorithms do not provide sufficient scalability. To overcome this problem, we introduce a new three-stage ME technique in Section IV.

In addition, our new technique works not only for the typical (pre-determined and fixed) MPEG GOP structures, but also for more general cases. This feature enables on-the-fly selection of GOP structures depending on the video content (e.g. detected scene changes, significant changes of motion, etc.). Furthermore, we introduce a new technique called multi-temporal approximation of MV fields (not to be confused with other forms of *multi-temporal* ME as found in H.264). These new techniques give more flexibility for a scalable MPEG encoding process.

In principle, any ME algorithm that works on the vector-selection or accuracy level can be inserted in our new scalable three-stage ME technique, because our proposal mainly scales on the structural level. Nevertheless, we prefer RME algorithms, since they explore temporal correlations in the video sequence and provide good motion vectors at a low number of candidate vector evaluations. However, state-of-the-art RME still performs unnecessary MV evaluations. For example, motion vectors are re-evaluated if candidate prediction vectors taken from different macroblocks are equal. This occurs if the currently processed block is a part of a larger area (e.g. the image background), from which the content in the enclosed blocks contains the same motion. To overcome this inefficiency, an enhanced RME algorithm is introduced in Section VI that forecloses excessive re-evaluation of identical MVs coming from different spatial vector predictions for the same block. Furthermore, the enhanced RME features a flexible number of processed macroblocks via block classification (see Section V), which provides even more scalability to the proposed ME system. Let us now first introduce the basic scalable three-stage ME.

IV. SMART, a structural level technique

A. Algorithm

Temporal candidate motion vectors play a key role within our scalable technique for ME. For this reason, we have adopted Recursive ME (RME), which is based on block matching. RME algorithms employ temporal candidates and provide a consistent ME in video sequences, while using a small number of candidate vector evaluations.

Obviously, the prediction quality of an ME algorithm improves with a smaller temporal distance D, where the parameter D denotes the difference between the frame numbers of the considered frames. Therefore, we commence with estimating the motion using the minimum temporal distance $|D| = 1$, which results in an accurate ME that can be performed at low computational effort. Since a common MPEG GOP-structure has $M > 1$ and thus some of the required vector fields must have $M > D > 1$, we consider this as a first stage to derive a prediction of vector fields. In a second stage, these predicted vector fields are used to calculate the required vector fields according to the MPEG standard (using larger D). In the third stage, the vector fields can be refined by using an additional — although simple — ME process. This stage is optional and only required if the highest quality has to be obtained (e.g. a conventional MPEG ME algorithm). Summarizing, our new concept results in a three-stage process, which is described more formally below.

- *Stage 1.* Prior to defining a GOP structure, we perform a simple RME for every consecutive frame X_n and compute the forward MV field $(X_{n-1} \rightarrow X_n)$ and then the backward field $(X_{n-1} \leftarrow X_n)$. For example, in Figure 2 this means computing vector fields like $\underline{f}_1^1$ and $\underline{b}_3^1$, but then for every pair of sequential frames (compare left side of Figure 3).
- *Stage 2.* After defining a GOP structure, all the vector fields $\underline{F} \in \{\underline{f}, \underline{b}\}$ required for MPEG encoding are approximated by appropriately accessing multiple

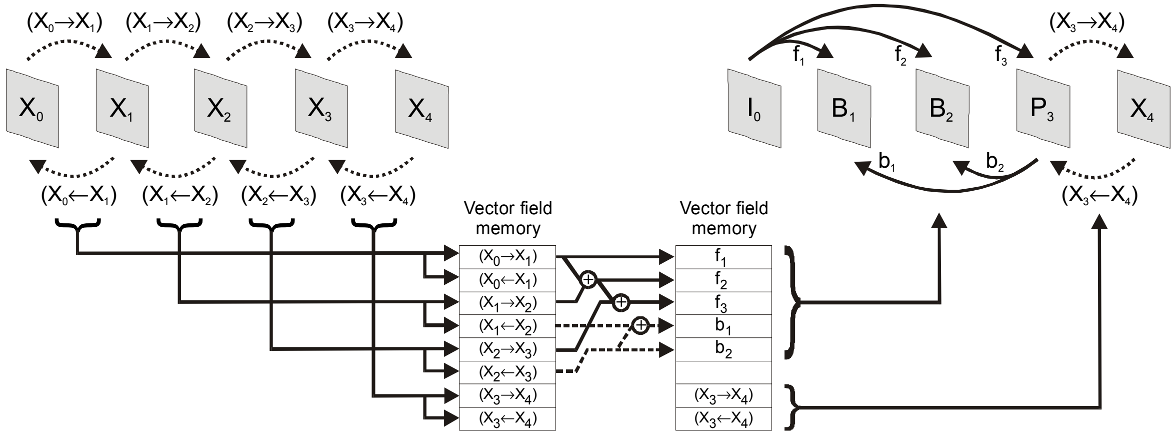

Fig. 3. Overview to the new motion estimation process. Vector fields are calculated for successive frames (left) and stored in memory. After defining the GOP structure, an approximation is calculated (middle) for the vector fields needed for MPEG coding (right).

available vector fields $\underline{F_A}$ and $\underline{F_B}$ and combine them using the linear relation

$$\underline{F} = \alpha * \underline{F_A} + \beta * \underline{F_A}, \qquad (1)$$

where the scaling factors α and β depend on the processed fields and are chosen according to the required temporal distance for $\underline{F}$. For example, $\underline{f_2} = (X_0 \rightarrow X_1) + (X_1 \rightarrow X_2)$ (see middle of Figure 3), thus having $\alpha = \beta = 1$. Note that α and β become different if the frame distances change or when complexity scaling is applied (see below).

- *Stage 3*. Optionally, a second iteration of RME is performed for refining the computed approximated MPEG vector fields from Stage 2. For example, the approximated vector fields from Stage 2 serve as temporal candidates in the finalizing refinement RME process in this last stage.

Rovati *et al.* [10] have proposed an approach that at first glance looks similar to the algorithm presented here. However, there are a number of important differences. Firstly, they initially estimate the MPEG vector fields and then process these fields for a second time, while keeping restricted to the MPEG GOP structure. This means that they have to deal with an increasing temporal distance to derive the vector fields already in the first step. This limits the accuracy of the computed first-step predictions. The processing of pictures in MPEG order is a second difference. Thirdly, the proposed ME does not provide scalability. The possibility of scaling vector fields, which are also used as multiple predictions, is mentioned in [10], but not further exploited. Our algorithm makes explicit use of this feature, which is a fourth difference. In the sequel, we explain important system aspects of our algorithm.

B. Architecture

Figure 4 shows the architecture of the SMART ME technique embedded in an MPEG encoder. Note that the amount of memory needed for the new architecture is the same as used for the architecture as shown in Figure 1, except for the additional motion vector memory. The memory costs are equal, because the entrance frame memory for Stage 1 can be integrated with the memory for the reordering process. The additional memory needed to store a vector field is negligible compared to the memory requirement of a video frame (a vector needs only 2 bytes vs. a luminance macroblock using 256 bytes). The three stages are decoupled from the actual coding process and are connected with each other via the central MV memory. This concept enables the processing of the three SMART stages in parallel, where results of one stage are reusable for another stage via the MV memory.

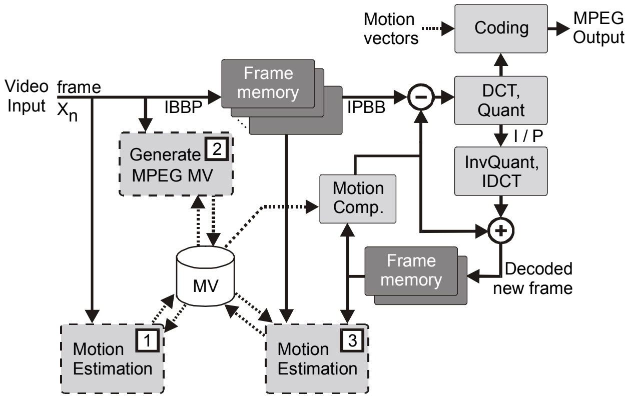

Fig. 4. Architecture of an MPEG encoder with the new scalable three-stage motion estimation technique.

Let us now discuss several architectural aspects of the individual stages of SMART.

- The initial ME process in Stage 1 is performed on succeeding frames ($|D| = 1$). Furthermore, Stage 1 uses original frames, without quantization errors. For these reasons, the RME yields a high-quality prediction with accurate MV fields.
- Stage 2 can optimally choose a GOP structure by e.g. analyzing the computed motion vector fields. For example, if motion occurs in a sequence, the first frame that initiates a group of frames having (almost) zero motion in-between can be defined as a reference frame.

On a sequence level, flexible scene-change detection can be added.

- With Equation (1), Stage 2 introduces a new concept called multi-temporal ME (a variant of this with the same name is used in H.264 coding). In our case, the term "multi-temporal" refers to two aspects. Firstly, the computation of Equation (1) means that one vector field is constructed from two other vector fields. Secondly, the total prediction of a vector field can be based on various vector fields such that several temporal references are used. The second aspect can be used for high-quality applications to approximate different real-life motions like video-object velocity, acceleration, zoom, rotation, etc. To give an example of multi-temporal ME for a video object with constant motion speed, we predict a MV field $\underline{\hat{f}}$ by specifying the motion model "most recent velocity" as

$$\underline{\hat{f}_i^k} = \begin{cases} 2 * \underline{f_{i-1}^k} - \underline{f_{i-2}^k} & \text{if } M_k \geq i \geq 3 \\ 2 * \underline{f_1^k} & \text{if } i = 2 \\ -\underline{b_{M_{k-1}}^{k-1}} & \text{if } i = 1, M_{k-1} > 1 \\ \underline{f_1^{k-1}} & \text{if } i = 1, M_{k-1} = 1, \end{cases} \tag{2}$$

where the term "most recent" refers to using the previously ($i-1$) computed vector field.[1]

C. Scalability aspects

The main advantage of the proposed SMART architecture is that it enables a broad scalability range of resource usage and achievable picture quality in the MPEG encoding process. This is illustrated by the following statements.

- Stage 1 and 3 can omit the computation of vector fields (e.g. the backward vector fields) or compute only significant parts of a vector field in order to reduce the computational effort and memory bandwidth.
- If the refinement in Stage 3 is omitted completely, the new technique can take advantage of further reduced computational effort, because the processing of vector fields in Stage 1 and 2 is much simpler than regular ME with the desired MPEG GOP ordering based on a large temporal distance.

D. Experimental verification

We show the scalability performance of the new technique with an initial experiment using the "Stefan" (tennis) sequence. The sequence is encoded based on a GOP size of $N = 12$ and $M = 4$ (thus "IBBBP" structure). We use the RME taken from [6] (limited to pixel-search) in Stage 1 and 3, because of its simple design. Note that the RME in Stage 1 used for this initial experiment could have been simpler, because of the minimum temporal distance at this stage. A scalable version of RME that performs equally to [6] but at much lower computational complexity is presented in Section VI.

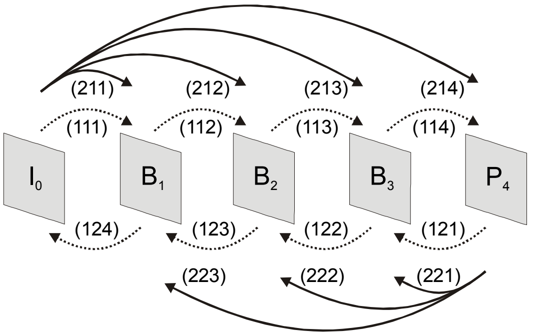

Fig. 5. Definition of priority order for motion vector fields for the experiment in Section IV-D. Dashed arrows are vector fields in Stage 1, solid arrows are refined MPEG vector fields in Stage 3.

In this experiment with $M = 4$, the number of vector fields (forward and backward motion) that are considered in an SGOP in Stage 1 is $2 * M = 8$ and in Stage 3 $M + (M-1) = 7$. To realize scalability, we gradually decrease the amount of vector field computations in Stage 1 and 3. To select vector fields for computation, we define a simple priority of the vector fields as follows. First, from the construction of the SMART ME technique it follows that Stage 1 is more important than Stage 3. Second, we consider forward motion as more important than backward motion, because P-frame predictions do not use backward motion. Third, MPEG vector fields in Stage 3 (or their equivalents in Stage 1) are considered less important with higher temporal distance (just for the sake of this experiment). This leads to vector field priorities as given in Figure 5. We do not compute vector fields such as $(I_0 \leftarrow B_1)$ in this experiment, because it is not required for computing the MPEG vector fields indicated with the large fat arrows in the figure. Note that in order to realize scalability and still keep track of the motion in Stage 1, the computation of vector fields is proceeding such that a new vector fields starts where a previous field has ended (there are no "holes").

The result of this scalability experiment is shown in Figure 6. The area with the white background is the quality range that results from scaling the computation complexity by varying the amount of computed motion-vector fields as described above. For comparison, we define the computation effort of the simple RME used by a standard MPEG encoder (which computes four forward vector fields and three backward vector fields per SGOP) as 100%, and use this as reference. Each vector field then requires 14% of the reference computation effort. If all vector fields of Stage 1 are computed and the refinement Stage 3 is performed completely, the computational effort is 200% (not optimized). Figure 6 shows that a large quality range is covered matching with the large differences in

[1] Note that besides our framework, any state-of-the-art motion estimation algorithm can be improved by using multi-temporal vector-field predictions. This implies that more than one prediction is generated for the computation of one vector field.

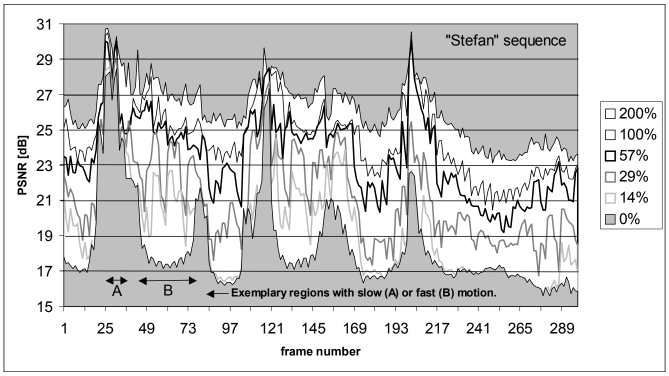

Fig. 6. PSNR of motion-compensated B-frames of the "Stefan" (tennis) sequence with different computational effort, P-frames are not shown for the sake of clarity (N=12, M=4). The percentage shows the different computational effort that results from omitting the computation of vector fields in Stage 1 or performing additional refinement in Stage 3.

computational effort.

V. Image block classification supporting ME

A. Introduction to alternative ME algorithms

Conventional ME algorithms process each image block in the same content-independent way. The drawback of these algorithms is that they spend many computations on calculating motion vectors for e.g. relatively flat blocks. Unfortunately, despite the effort, the ME process yields MVs of poor quality. Block classification is a known technique for concentrating the ME on blocks that may lead to accurate motion vectors [12].

In a further step, the classification can be controlled such that the number of blocks that are allowed for evaluation leads to complexity scalability. A recent publication on scalable motion estimation using block classification was presented in [13]. However, the classification provides blocks for which a good motion vector should be found, but the classification is only used as a binary decision between structured blocks with a fixed set of vector candidates and non-structured blocks that can be skipped in the ME process.

In our paper, the usage of the structure in the blocks is realized by providing the ME algorithm with the information whether it is more likely to find a good motion vector in up-down or left-right search directions. This information is derived from a simple classification algorithm described below, which is based on detecting horizontal and vertical transitions (edges). Since ME will find equally good motion vectors for every position *along* such an edge (where a displacement in this direction does not introduce large displacement errors), searching of motion vectors *across* this edge will rapidly reduce the displacement error and thus lead to an appropriate motion vector. Horizontal and vertical edges can be detected by significant changes of pixel values in vertical and horizontal direction, respectively.

B. Block-classification using edge-detection algorithm

The edge-detecting algorithm we use is in principle based on continuously summing up pixel differences along rows or columns and count how often the sum exceeds a certain threshold. Let p_i with $i = 0,1,..,15$ the pixel values in a row or column of a macroblock (size 16x16). We then define a range where pixel divergence (expressed as l_i) is considered as noise if $|l_i|$ is below a threshold t. The pixel divergence is defined by the Table I.

TABLE I
Definition of pixel divergence, where the divergence is considered as noise if it is below a certain threshold.

Condition	Level l_i
$i=0$	0
$i=1..15 \wedge \lvert l_{i-1} \rvert \le t$	$l_{i-1}+(p_i-p_{i-1})$
$i=1..15 \wedge \lvert l_{i-1} \rvert > t$	$l_{i-1}+(p_i-p_{i-1})-\text{sgn}(l_{i-1})*t$

The area preceding the edge yields a level in the interval $[-t;+t]$. The middle of this interval is at $l = 0$, which is modified by adding $\pm t$ for the case that $|l|$ exceeds the interval around zero (start of the edge). This mechanism will follow the edges and prevent noise from being counted as edges. The counter c as defined below indicates how often the actual interval was exceeded.

$$c = \sum_{i=1}^{15} \begin{cases} 0 & \text{if } |l_i| \le t \\ 1 & \text{if } |l_i| > t \end{cases} \tag{3}$$

The occurrence of an edge is defined by the resulting value of c from Equation 3.

This edge-detecting algorithm is scalable by selecting the threshold t, the number of rows and columns that are considered for the classification, and a typical value for c. A block is considered for class "horizontal edge" or "vertical edge" if a clear edge is found for the column or row test, respectively. A clear edge should exceed a typical value for c. Obviously, we can derive from the previous equations the two extra classes "flat" for all blocks that do not belong to "horizontal edge" and "vertical edge", and the class "diagonal/structured" for blocks that belong to both classes "horizontal edge" and "vertical edge" simultaneously. We have adopted this edge-detection technique because of its simplicity and its suitability for block classification.

C. Experimental verification

Experimental evidence has shown that in spite of the complexity scalability of this classification algorithm, the evaluation of a single row or column in the middle of a picture block was found sufficient for a rather good classification. Figure 7 shows the result of an example to classify image blocks of size 16x16 pixels (macroblock size). For this experiment, a threshold of $t = 25$ was used. We considered a block to be classified as "horizontal edge" if $c \ge 2$ holds for the central column computation and class "vertical edge" if

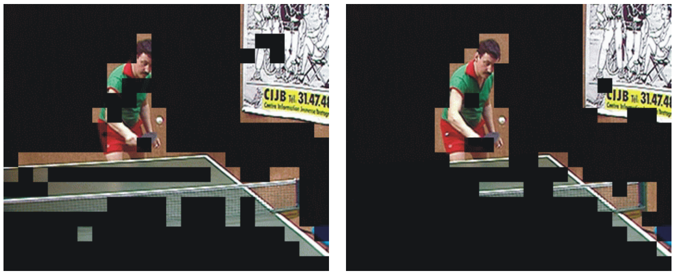

Fig. 7. Visualization of block classification using a picture of the "Tennis table" sequence. The left (right) picture shows blocks where horizontal (vertical) edges are detected. Blocks that are visible in both pictures belong to the class "diagonal/structured", whereas blocks that are blanked out in both pictures are considered as "flat".

$c \geq 2$ holds for the row computation.

The presented block classification will be used to support an alternative ME algorithm presented in the next section. The emphasis of the alternative algorithm is on a clever selection of motion vectors for evaluation, whereas the SMART technique from Section IV is a structure-level technique.

VI. CARES, a Vector-Selection level ME technique

In this section, a new vector-selection level concept for a scalable RME is presented, that can be integrated into SMART, the three-stage ME technique on the structural level (see Section II), as a more advanced replacement for the simple RME that is used in Stage 1 and 3 of SMART. The block-classification algorithm (see Section V) is designed to support the ME algorithm and is thus is a key feature in this new concept. The block classification is used to concentrate the ME on blocks that should lead to good motion vectors, whereas the remaining "flat" blocks become a MV assigned without any further MV evaluation. This idea was adopted from [13]. However, except for the difference in using the structure of the blocks (see Section V), another difference is that good vector candidates are proposed to "future" blocks such as the neighboring blocks at the right and just below the actual block. Therefore, our algorithm is more suited for software-oriented implementations. Further system aspects are discussed at the end of this section.

A second key feature of the new concept is a more advanced MV prediction, as compared to conventional RME. In state-of-the-art RME, the set of candidate motion vectors that are evaluated to find the displacement of a certain macroblock contains a few vector candidates, which are adopted from already processed macroblocks (the vectors are queried from a MV field memory that stores the finally selected MVs for the already processed blocks). This prediction mechanism leads to re-evaluation of the same motion vector for a block, if the block is a part of a larger equally moving area (e.g. the image background). Instead, we distribute certain motion vectors of good accuracy from actually processed blocks to neighboring surrounding blocks, which are then triggered to evaluate the distributed vector candidates. This may lead to a series of vector evaluations, but in practice the total amount of evaluated vectors is scalable and lower than with conventional RME. The MVs that yield less accuracy for ME are not distributed.

A. Algorithm

The proposed motion estimation algorithm called CARES (Content-Adaptive REcursive Scalable) ME, is as follows.

1. Using the block classification presented in Section V, two lists L_+ and L_- are created containing macroblocks that are classified as having horizontal or vertical edges (L_+) or being flat (L_-).
2. All macroblocks $mb_+ \in L_+$ are initialized with an approximated motion vector (temporal candidate) if available, or with the zero vector otherwise.
3. Based on the best motion vector mv_b found so far for the current macroblock $mb_+ \in L_+$, the following motion vector candidates $mv_c = (dy, dx)$ are tested.
   - $mv_c = (-1,0)$ and $mv_c = (+1,0)$ for macroblocks having a horizontal edge.
   - $mv_c = (0,-1)$ and $mv_c = (0,+1)$ for macroblocks having a vertical edge.

   If a new best motion vector mv_b^* is found for the current macroblock, this vector is proposed to other macroblocks by inserting the eight surrounding macroblocks of the current macroblock into a temporary list L_{tl}. The macroblocks that are inserted to this list are restricted to be from list L_+, thus $L_{tl} \subseteq L_+$. The macroblock of list L_{tl} are then processed in the following step.
4. If L_{tl} is not empty, all macroblocks $mb_{tl} \in L_{tl}$ evaluate the vector mv_b^* of Step 3 as candidate vector. Each macroblock $mb_{tl} \in L_{tl}$ further distributes vector mv_b^* by inserting its adjacent macroblocks into L_{tl}, if they belong to list L_+ and if mv_b^* is a better motion vector than the current best motion vector of mb_{tl}. This step is repeated until L_{tl} is empty.
5. If L_+ is not empty, the next macroblock mb_+ in list L_+ is processed with Step 3.
6. Finally, all macroblocks $mb_- \in L_-$ get the MV of one of its neighbors mb_n, if $mb_n \in L_+$, or the zero vector as default. In both cases, the vector is not further evaluated.

B. Experimental verification

An experiment has been set up to compare the CARES ME

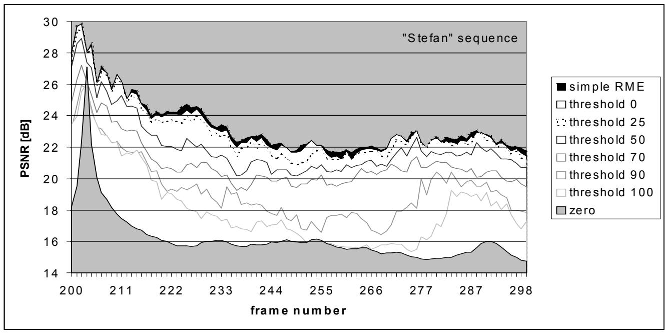

Fig. 8. Comparison of simple and scalable RME. The frame number refers to a part of the "Stefan" sequence with "IPPP" structure, where the PSNR differences are best visible. The fat black curve areas indicate the quality gap between simple RME and CARES ME. The white area is the complexity scalability range of CARES ME.

of this section with the simple RME [6] that was used within the SMART technique in Section IV. We compared the measured PSNR of the motion-compensated frames (based on a GOP size of $N = 12$ and $M = 1$, thus "IPPPP" structure) of the "Stefan" sequence, when using the simple RME or the CARES ME. The block-classification algorithm (see Section V) was used with different thresholds. Figure 8 shows the result of this experiment. The figure shows the large scalability range between 16 dB and 24 dB PSNR, just by variation of the threshold t. By decreasing the classification threshold to a lower number, the quality comes arbitrary close to simple RME. A typical setting of $t = 25$ results in a PSNR that is rather close to the simple RME.

The average PSNR and the average number of MV evaluations per macroblock for different sequences are shown in Figure 9. It can be seen that for the typical threshold setting of $t = 25$, the obtained PSNR is within 1 dB from the simple RME algorithm. However, the simple RME uses 18.35 vector evaluations on the average, whereas CARES requires only between 1.9 and 4.9 vector evaluations, depending on the sequence. In the worst-case situation using a threshold of $t = 0$ (all macroblocks are processed), the same quality as with simple RME is obtained. For example, even for the worst-case "Stefan" sequence, the computational complexity is reduced by at least 58%, resulting in an average of 7.78 MV evaluations per macroblock. The "Tennis table" sequence, which has less motion, needs less than 1 vector evaluation per macroblock, leading to medium quality. The complexity is smoothly scalable until zero-vector fields are reached.

VII. ME SYSTEM EXPERIMENTS AND RESULTS

A. Experimental set-up

In Sections IV to VI, we presented the novel ME techniques SMART (structural level) with frame processing in display order and the CARES ME algorithm (vector-selection level), based on block classification. Both techniques have been combined and integrated in an MPEG coder for measuring their scalability performance.

We experimented with different sequences and we scaled the computational complexity with two parameters. The first parameter defines the number of motion vector fields that are computed per SGOP (we use a GOP size of $N = 12$ and $M = 4$, thus "IBBBP" structure). The computation order of vector fields was identical to the evaluation of the SMART technique in Section IV-D. Summarizing, vector fields from Stage 1 are computed before vector fields from Stage 3, forward motion before backward motion and fields with shorter temporal distances are computed before fields with larger temporal distances. The priority order of MPEG vector fields to be refined in Stage 3 is again $\underline{f_1}$, $\underline{f_2}$, $\underline{f_3}$, $\underline{f_4}$, $\underline{b_3}$, $\underline{b_2}$, $\underline{b_1}$. The priority order of the vector fields in Stage 1 is defined such that the vector fields successively cover the prediction depth of the SGOP starting from frame $(I/P)_{k*M}$ to $(I/P)_{(k+1)*M}$ for the M forward fields and then vice-versa for the M backward fields. The second parameter for achieving scalability varies the classification threshold t of the block classification used in the CARES ME algorithm.

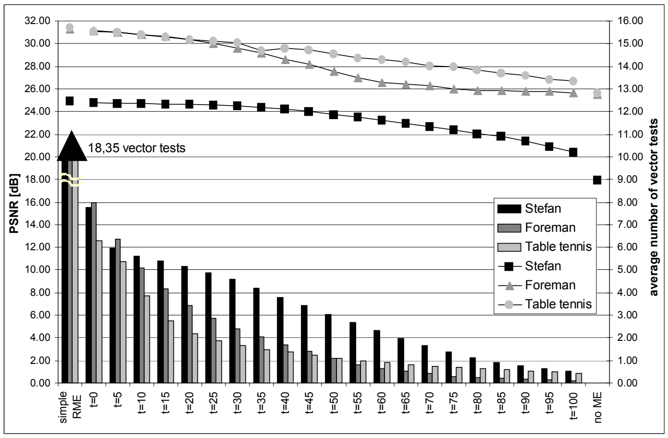

Fig. 9. Comparison of simple RME and CARES ME. The top curves show the average PSNR for three different video sequences. The bars show the average number of vector evaluations per macroblock.

B. Results

Figure 10 shows the average number of vector candidates that were evaluated for each macroblock of the computed MPEG motion vector fields. The priority level (indicated by gray levels) refers to the number of computed vector fields as given by the priority order used for SMART, and the indicated threshold in the horizontal direction is the same as used in CARES. The figure shows that the MV evaluations scale smoothly with the threshold and the number of computed vector fields. Note that the choice of a small non-zero threshold already leads to a significant reduction of the average number of vector evaluations.

The average achieved PSNR of the predicted P- and B-frames (taken after motion compensation and before computing the differential signal) is shown in Figure 11. For a full-quality comparison, we consider full-search block matching with a search window of 32x32 pixels. The new joint

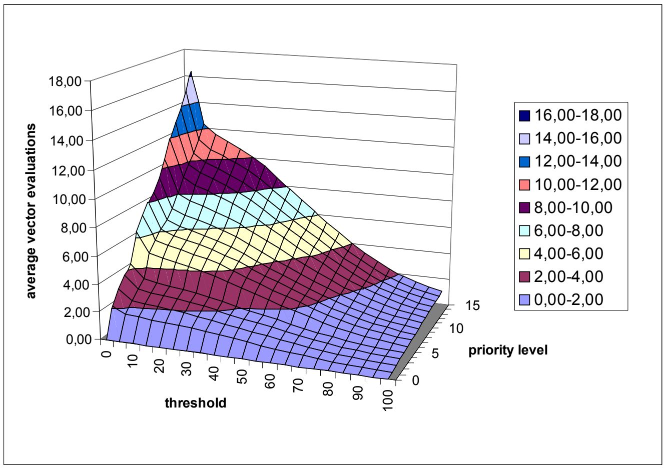

Fig. 10. Statistics of the average number of motion vector evaluations performed per macroblock for the "Stefan" sequence.

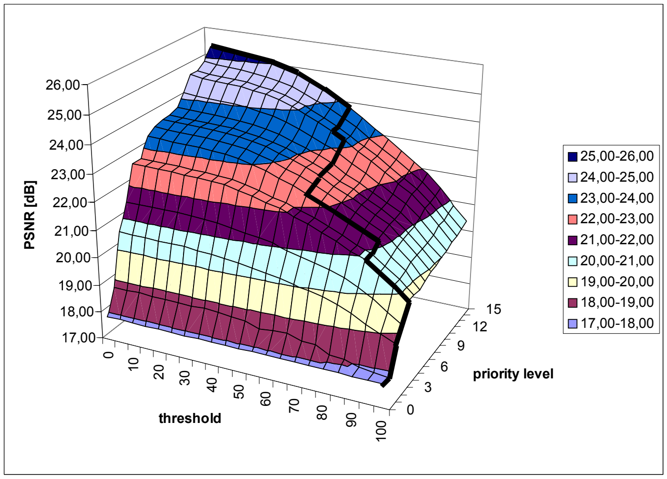

Fig. 11. PSNR of the motion-compensated frames when processing the "Stefan" sequence. The y-axis indicates the number of vector fields that are computed per SGOP and the x-axis is the threshold for the CARES ME algorithm. The fat black line on the surface follows the path of the highest PSNR.

TABLE II
AVERAGE PSNR OF THE MOTION-COMPENSATED P- AND B-FRAMES OF "STEFAN" FOR DIFFERENT ME ALGORITHMS (SEE THE TEXT FOR MORE DETAIL).

Algorithm	Tests/MV	PSNR	Scalable ME
2DFS (32x32)	923.52	24.92	9.42
NTSS [3]	25.13	22.63	2.69
Diamond [4]	21.77	22.53	2.31
Scalable ME	see last column		

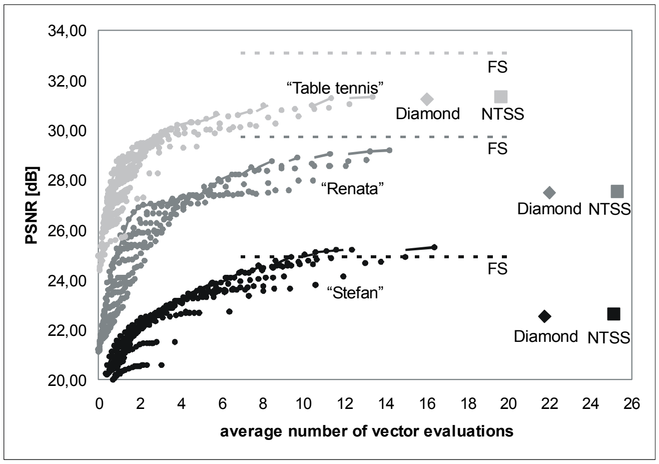

Fig. 12. Performance comparison of the scalable ME system (dots = all configurations, solid line = optimum) with diamond search (diamond) and NTSS (square) using the "Stefan" sequence (black), "Renata" (medium gray) and "Table tennis" (light gray). The dashed line indicates the PSNR that is obtained with a 32x32 FS.

ME technique slightly outperforms full search by up to 0.39 dB PSNR measured from the motion-compensated P- and B-frames of this experiment (25.31 dB instead of 24.92 dB).

Further comparisons were made with the scalable ME system and alternative fast state-of-the-art ME algorithms. Table II shows the average PSNR of the motion-compensated P- and B-frames for the "Stefan" sequence and four ME algorithms with the same conditions as described above (same N, M, etc.). The first data column (tests per MV) shows the average number of vector evaluations that were performed per macroblock to indicate the computational performance of the algorithms. The last column ("scalable ME") contains the average number of vector evaluations required for our scalable ME system (at optimal configuration for this experiment) to reach the same quality as the ME algorithms for comparison. Note that MV evaluations pointing outside the picture are not counted, which results in numbers that are lower than the nominal values (e.g. 923.52 instead of 1024 for 32x32 full-search (FS)). It can be seen that the scalable ME system requires only a fraction of MV evaluations that are performed by the compared ME algorithms to reach the same quality provided by these algorithms.

A quality comparison expressed in PSNR for more video sequences is shown in Figure 12, where we used the sequence "Stefan" with high motion, and "Foreman" and "Table Tennis" having low motion. The curves in the figure show a waterfall effect, which means that a fast quality build-up is realized at the beginning of the scalability range, and that there is room for further improvements on the quality at the end of the scalability range. The bending point of the waterfall is between 3 and 5 vector evaluations per block. It can also be seen that for more complex scenes, the scalable ME system obtains the quality of FS at about 10-16 vector evaluations per block. For the "Table tennis" sequence, it seems that the provided scalability range does not reach the full-search level. However, because the absolute quality level is already much higher than for more critical scenes, we did not examine this further.

C. Discussion

At this point, we want to address some system aspects. First, we already explained that the scalable ME system has been based on a more software oriented implementation. This can be seen from the fact that in the CARES ME, we first concentrate on blocks with a structured detail giving good vector candidates, and we simply assign vectors to the remaining blocks. Second, we provide good vector candidates to all surrounding blocks also at the right and below the current block position. For this reason, the acceptation process for a good vector can distribute itself within the frame in various directions. In order to implement this, a cache memory will be required of one frame in the worst case. However, we expect that in most cases (given the usual size of objects), a much smaller cache can be used without sacrificing too much quality.

We have adopted the software-oriented approach, because the scalability range was implemented towards the direction of portable devices having small computational resources. The architecture of such devices is usually based on the combination of a DSP processor with a programmable RISC core, so that applications are software-oriented in any case.

VIII. CONCLUSIONS

We have presented new scalable ME techniques for MPEG encoding, which are important for realizing portable MPEG video applications in devices where computing power is limited. The computational complexity scalability is obtained by scaling the number of processed motion-vector (MV) fields and the number of vector evaluations. The first term relates to the GOP structure in MPEG, whereas a second term relates to the clever selection of MVs for computation. It has been shown that the combination of both techniques into one scalable ME system can reduce the computational effort over a large range, making our system feasible for low-cost mobile MPEG systems.

The first ME technique selecting MV fields, called SMART, has been split into a pre-computation stage and an approximation stage. Optionally, a refinement stage can be added to approach the quality of a conventional MPEG encoder (or even outperform it). In the pre-computation stage, we used RME to find rather good motion estimates because the frames are processed in time-consecutive order. In the approximation stage, MV-fields are scaled and added or subtracted (thus having multi-temporal references) to come to the finally required MPEG MV-fields. These computations are much simpler than performing advanced vector searches. The computation of e.g. the backward ME can be omitted to save computational effort and memory bandwidth usage. The optional third refinement stage runs another small RME process based on the vector-fields approximations of the second stage.

The second technique addressing vector selection, called CARES, proposes a new scalable RME for integration into the above-mentioned SMART technique. The scalable RME is based on a simple block classification that provides information about the occurrence of horizontal and/or vertical edges in the block content. This information is then subsequently used to control the prediction of good MV candidates. This feature leads to a reduced set of motion vector candidates that are evaluated in comparison with state-of-the-art ME algorithms for MPEG applications. The CARES ME algorithm proposes good MVs to surrounding blocks for further evaluation. This extension prevents the re-evaluation of identical MVs for a block, which regularly occurs in state-of-the-art RME. The disadvantage is that a significant cache memory will be required.

Experiments with our new ME system using the "Stefan" sequence show that a full processing of our framework compares well in picture quality (PSNR) with a 32x32 full-search ME (or even outperform it). When compared to existing fast ME algorithms published for MPEG, our system requires roughly 30% to 90% less MV evaluations for different sequences from less motion to high motion, when the system was scaled to obtain the same quality of the fast algorithms. Since the number of MV evaluations relates to the computational complexity of the ME process that requires a significant part of the computational effort in MPEG encoding, the overall encoder complexity is considerably reduced.

The complexity scalability of the complete ME system proposal is between below 1 up to 15 vector evaluations per macroblock on the average, leading to a global PSNR variation of 7.0 to 7.4 dB PSNR of the motion-compensated frames. The obtained computational complexity scalability is seen sufficient for a large range of stationary and portable MPEG coding applications.

REFERENCES

[1] C. Hentschel, R. Braspenning and M. Gabrani, “Scalable Algorithms for Media Processing,” Proceedings of IEEE International Conference on Image Processing (ICIP), vol. 3, pp 342-345, October 2001.

[2] R.J. Bril, C. Hentschel, E.F. M. Steffens, M. Gabrani, G. van Loo, J.H.A. Gelissen, “Multimedia QoS in Consumer Terminals,” Proceedings of IEEE International Workshop on Signal Processing Systems (SIPS), pp. 332-343, September 2001.

[3] Reoxiang Li, Bing Zeng and M.L. Liou, “A new Three-Step Search Algorithm for Block Motion Estimation,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 4, no. 4, pp. 438-442, August 1994.

[4] Jo Yew Tham, S. Ranganath, M. Ranganath and A.A. Kassim, “A Novel Unrestricted Center-Biased Diamond Search Algorithm for Block Motion Estimation,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 8, no. 4, pp. 369-377, August 1998.

[5] G. de Haan, P.W.A.C. Biezen, H. Huijgen and O.A. Ojo, “True-Motion Estimation with 3-D Recursive Search Block Matching,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 3, no. 5, pp. 368-379, October 1993.

[6] P.H.N. de With, “A Simple Recursive Motion Estimation Technique for Compression of HDTV Signals,” Proceedings of IEE International Conference on Image Processing and its Applications (IPA), pp. 417-420, 1992.

[7] Mei-Juan Chen, Liang-Gee Chen, Tzi-Dar Chiueh and Yung-Pin Lee, “A new Block-Matching Criterion for Motion Estimation and its Implementation,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 5, no. 3, pp. 231-236, June 1995.

[8] K. Lengwehasatit, A. Ortega, A. Basso and A. Reibman, “A Novel Computationally Scalable Algorithm for Motion Estimation,” Proceedings on SPIE International Conference on Visual Communications and Image Processing (VCIP), pp. 68-79, January 1998.

[9] R. Braspenning and G. de Haan, “Effcient Motion Estimation with Content-Adaptive Resolution,” Proceedings of International Symposium on Consumer Electronics (ISCE), pp. E29-E34, September 2002.

[10] F.S. Rovati, D. Pau, E. Piccinelli, L. Pezzoni and J.-M. Bard, "An Innovative, High Quality and Search Window Independent Motion Estimation Algorithm and Architecture for MPEG-2 Encoding," IEEE Transactions on Consumer Electronics, vol. 46, no. 3, pp. 697-705, August 2000.

[11] G. de Haan and P.W.A.C. Biezen, "Sub-pixel motion estimation with 3-D recursive search block-matching," Signal Processing: Image Communication, vol. 6, pp. 229-239, 1994.

[12] T. Kummerow and P. Mohr, "Method of Determining Motion Vectors for the Transmission of Digital Picture Information," EP 0 496 051, November 1991, European Patent Application.

[13] R. Braspenning, G. de Haan and C. Hentschel, "Complexity Scalable Motion Estimation," Proceedings of SPIE Visual Communications and Image Processing, vol. 4671(1/2), pp. 442-453, 2002.

Stephan Mietens was born in Frankfurt/Main, Germany, in 1972. He graduated in computer science from the Technical University of Darmstadt, Germany, in 1998 on the topic of "Asynchronous VLSI design". Subsequently, he joined the University of Mannheim, where he started his research on "Flexible Video Coding and Architectures" in cooperation with Philips Research Laboratories in Eindhoven, The Netherlands. He joined the Eindhoven University of Technology in Eindhoven, The Netherlands, in 2000, where he is working towards a Ph.D. degree on "Scalable Video Systems". Since 2003 he became a scientific researcher at Philips Research Labs. Eindhoven, The Netherlands, in the Storage and System Applications group, where he is involved in projects to develop new coding techniques.

Peter H.N. de With graduated in electrical engineering from the University of Technology in Eindhoven. In 1992, he received his Ph.D. degree from the University of Technology Delft, The Netherlands, for his work on video bit-rate reduction for recording applications. He joined Philips Research Labs Eindhoven in 1984, where he became a member of the Magnetic Recording Systems Department. From 1985 to 1993 he was involved in several European projects on SDTV and HDTV recording. In this period he contributed as a coding expert to the DV standardization for digital camcording. In 1994 he joined the TV Systems group, where he was leading the design of advanced programmable video architectures. In 1996, he became senior TV systems architect and in 1997, he was appointed as full professor at the University of Mannheim, Germany, at the faculty Computer Engineering. In 2000, he joined CMG Eindhoven as a principal consultant and he became professor at the University of Technology Eindhoven, at the faculty of Electrical Engineering. He has written numerous papers on video coding, architectures and their realization. Regularly, he is a teacher of the Philips Technical Training Centre and for other post-academic courses. In 1995 and 2000, he co-authored papers that received the IEEE CES Transactions Paper Award. In 1996, he obtained a company Invention Award. In 1997, Philips received the ITVA Award for its contributions to the DV standard. Mr. de With is a senior member of the IEEE, program committee member of the IEEE CES, chairman of the Benelux community for Information and Communication Theory and board member of various working groups.

Christian Hentschel (M'99) received his Dr.-Ing. (Ph.D.) in 1989 and Dr.-Ing. habil. in 1996 at the University of Technology in Braunschweig, Germany. He worked on digital video signal processing with focus on quality improvement. In 1995, he joined Philips Research in Briarcliff Manor, USA, where he headed a research project on moiré analysis and suppression for CRT based displays. In 1997, he moved to Philips Research in Eindhoven, Netherlands, leading a cluster for Programmable Video Architectures. He got the position of a Principal Scientist and coordinated a project on scalable media processing with dynamic resource control between different research laboratories. Since August 2003, he is a full professor at the University of Technology in Cottbus, Germany, where he heads the department of Media Technology. He is a member of the Technical Committee of the International Conference on Consumer Electronics (IEEE) and a member of the FKTG in Germany.