Multi-Frame Affine Motion-Compensated Prediction for Video Compression

Thomas Wiegand[1], Eckehard Steinbach[1], and Bernd Girod[1,2]

[1] Telecommunications Laboratory
University of Erlangen-Nuremberg
Erlangen, Germany
E-Mail: [wiegand,steinb]@lnt.de

[2] Information Systems Laboratory
Stanford University
Stanford, CA, USA
girod@ee.stanford.edu

Abstract

Multi-frame affine prediction extends motion compensation from the previous frame to several past decoded frames and warped versions thereof. Affine motion parameters describe the warping. In contrast to translational motion compensation, the affine motion parameters must be assigned to large image segments to obtain a rate-distortion efficient motion representation. These large image segments usually can not be chosen so as to partition the image uniformly. Hence, encoding proceeds in four steps: (*i*) estimation of several affine motion parameter sets between the current and previous frames, (*ii*) generating the multi-frame buffer consisting of past decoded frames and affine warped frames, (*iii*) multi-frame block-based hybrid video encoding, and (*iv*) determination of the efficient number of motion models using Lagrangian optimization techniques. A significant improvement in coding efficiency can be observed when comparing the multi-frame affine motion coder to the TMN-10 coder, the rate-distortion optimized test model of the H.263 standard. At a fixed quality of 34 dB PSNR, the proposed coder achieves 24 % bit-rate reduction over a set of 8 different test sequences. The bit-rate savings inside this set of test sequences vary from 35 % to 15 % which correspond to PSNR gains of 3 dB and 0.8 dB, respectively. It is shown that both concepts, affine motion and long-term memory prediction, contribute to the overall gain.

I. Introduction

In recent years, significant gains in video compression have been achieved by increasing the adaptability of the compression schemes, mainly on the motion compensation side [1]. Hence, video compression schemes have "learned" to handle more and more sophisticated scenes. This paper introduces a new motion representation scheme that is suitable for the compression of sequences that are difficult to code with state-of-the-art video compression designs like H.263+ [2].

Hybrid video codecs are one of the most successful class of today's video compression designs. The concept of block-based motion-compensated prediction (MCP) is prevalent in all these coding schemes [1]. The achievable MCP performance can be increased by reducing the size of the motion-compensated blocks [3]. This is taken into account, for example, by the INTER-4V mode of H.263 [2] where instead of a block of size 16×16 luminance pixels four blocks of size 8×8 luminance pixels are employed. However, the bit-rate must be assigned carefully to the motion vectors of these smaller blocks. Therefore, rate-constrained motion estimation is often employed yielding improved compression efficiency [3], [4], [1]. In rate-constrained motion estimation, a Lagrangian cost function $J = D + \lambda R$ is minimized, where distortion D is weighted against rate R using a Lagrange multiplier λ. Moreover, also the macroblock mode decision should be based on Lagrangian optimization techniques [5]. These facts have been recognized within the ITU-T/SG16/Q15 group when adopting TMN-10, the test model of the H.263 standard [6] which is based on Lagrangian optimization techniques [7].

For multi-frame prediction referencing past decoded frames, the name long-term memory prediction has been coined [8]. Long-term MCP increases the efficiency of video compression schemes by utilizing several past frames that are assembled in a multi-frame buffer. This buffer is simultaneously maintained at encoder and decoder. Block-based motion compensation is performed using motion vectors that consist of a spatial displacement and a picture reference parameter to address a block in the multi-frame buffer. Rate-constrained motion estimation is used to control the bit-rate of the motion and picture reference parameters. In [8], bit-rate savings between 10 % and 15 % are reported when comparing a 10 frame long-term memory video coder to TMN-10, the test model of the H.263 standard. The ITU-T/SG16/Q15 group has decided to adopt this feature as an Annex to the H.263 standard [9].

While, long-term memory prediction extends the motion model to exploit long-term dependencies in the video sequence, the motion model remains translational. Independently moving objects in combination with camera motion and focal length change lead to a sophisticated motion vector field which may not be efficiently approximated by a translational motion model. With an increasing time interval between video frames, this effect is further enhanced since more complex motion is likely to occur. Hence, the translational motion model may no longer be the best choice for the approximation of the motion vector field.

Tsai and Huang derive a parametric motion model that relates the motion of planar objects to the observable motion field in the image plane [10]. This model is often referred to as the bilinear motion model and the parameters are estimated using corresponding points [10]. In [11], Hötter and Thoma estimate the eight parameters of the bilinear motion model using spatial and temporal intensity gradients in the context of object-based video coding. Diehl exploits in [12] affine (6 degrees of freedom) and bilinear motion models for object-based video coding. Various other researchers have utilized affine and bilinear motion models to represent the image plane motion of video objects or regions [13], [14], [15], [16], [17]. The motion coefficients are estimated such that they lead to an efficient representation of the motion field inside the corresponding image partition. Due to the mutual dependency of motion estimation and image partition a combined estimation must be utilized. This results in a sophisticated optimization task which usually is very time consuming. Moreover, providing the encoder the freedom to specify a precise segmentation has generally not yet resulted in a significant improvement of compression performance for natural camera-view scene content due to the number of bits needed to specify the segmentation.

Other researchers have used affine or bilinear motion models in conjunction with blocks which are prevalent in all video coding standards to obtain rate-distortion efficient MCP [18], [19], [20]. They have faced the problem that especially at low bit-rates the overhead associated with higher order motion models that are assigned to smaller size blocks might be prohibitive. A combination of the block-based and the region-based approach is presented in [21]. Karczewicz et al. show in [21] that the use of polynomial motion models in conjunction with a coarse segmentation of the video frame into regions, that consist of a set of connected blocks of size 8×8 pixels, can be beneficial in terms of coding efficiency.

The methodology in this work differs from previous approaches in that the motion models are not associated with a particular image segment. Although the motion coefficients are determined on a finite sub-area of the image, the model can be employed at any position inside the frame. An efficient realization of this segmentation-free motion representation is the use of multiple warped reference frames for prediction. Instead of performing a joint estimation of the image partition and the associated motion models, we warp reference frames and select these in a rate-distortion efficient way on a block basis. Thus, warped reference frames are selected in the same way as past reference frames in long-term memory MCP. Hence, we have transformed the joint optimization task of finding an efficient combination of motion models, regions and other parameters to separate steps. Each of these steps takes an almost constant amount of computation time which is independent of the context of the input data. The coder robustly adapts the use of the affine motion models to the input statistics and never degrades below the rate-distortion performance that can be achieved with the syntax of the underlying H.263 standard. The use of multiple reference frames requires only very minor syntax changes of state-of-the-art video coding standards.

The idea of warping reference frames has been published by the authors already in [22] where multiple warped versions of the previously decoded frame are employed for multi-frame block-based MCP. Note that in [23], a similar approach has appeared that was developed independently to our work [22]. Our scheme [22] is extended by a more efficient affine motion estimation algorithm that improves motion compensation performance and computational complexity in [24]. Additionally, the scheme in [22], [24] is extended by long-term memory prediction in that warped frames and block-based motion compensation can be conducted by referencing several past frames [25].

This paper is organized as follows. First, the proposed multi-frame video coding architecture and syntax is presented followed by the affine motion model. Second the coder control is explained. The estimation procedure for the affine motion parameters and the reference picture warping are described. Then, the incorporation of the affine motion approach into a rate-constrained motion estimation and mode decision framework is explained. Finally, experimental results are presented that illustrate the improved rate-distortion performance of the proposed coder in comparison to TMN-10.

II. Affine Multi-Frame Motion-Compensated Prediction

The architecture of the multi-frame affine motion-compensated predictor is depicted in Fig. 1. The

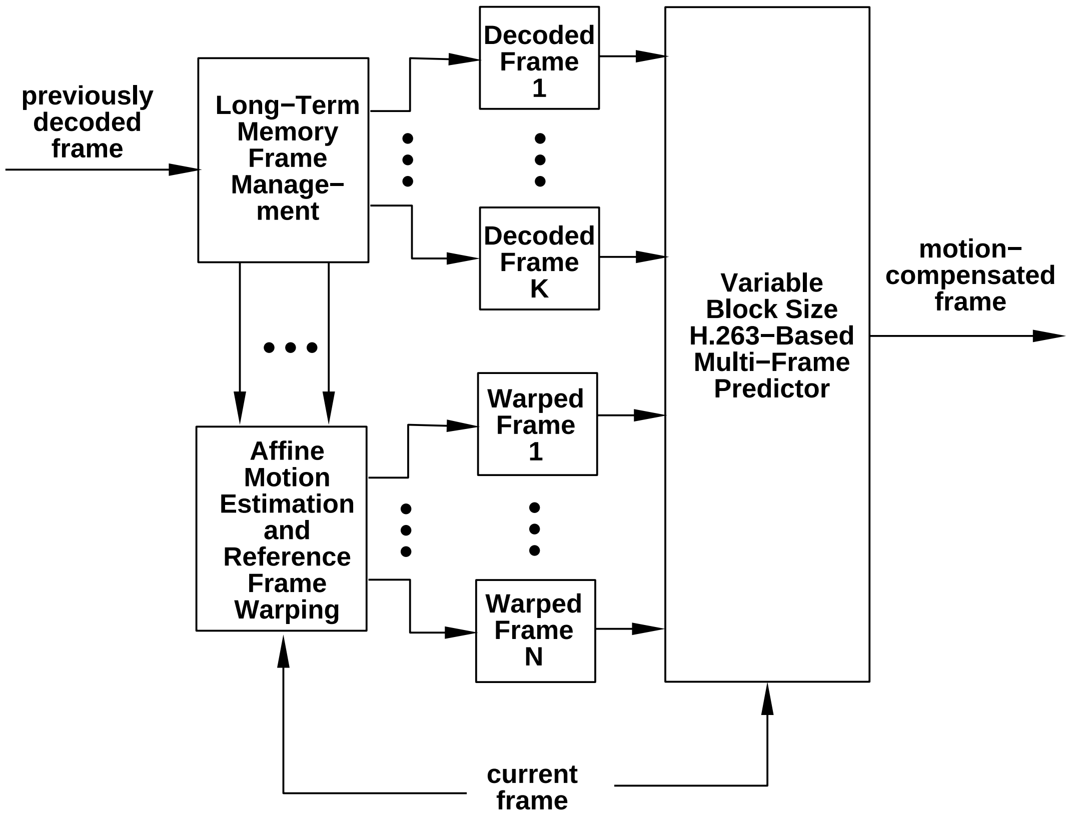
previously decoded frame
Long−Term Memory Frame Manage−ment
Decoded Frame 1
Decoded Frame K
Warped Frame 1
Warped Frame N
Affine Motion Estimation and Reference Frame Warping
Variable Block Size H.263−Based Multi−Frame Predictor
motion−compensated frame
current frame

Fig. 1. Architecture of the affine multi-frame motion-compensated predictor.

motion-compensated predictor utilizes $M = K + N$ $(M \geq 1)$ picture memories. The M picture memories are composed of two sets:

- K past decoded frames and
- N warped versions of past decoded frames.

The H.263-based multi-frame predictor conducts block-based MCP using all $M = K + N$ frames and produces a motion-compensated frame. This motion-compensated frame is then used in a standard hybrid DCT video coder [1], [2]. The N warped reference frames are determined using the following two steps.

1. Estimation of N affine motion parameter sets between the K previous frames and the current frame.
2. Affine warping of N reference frames.

The effective number of reference frames $M^* \leq M$ is determined by evaluating their rate-distortion efficiency in terms of Lagrangian costs for each reference frame. The M^* chosen reference frames with the associated affine warping parameters are transmitted in the header of each picture. The order of their transmission provides a model index that is used to specify a particular reference frame on the block basis. The decoder maintains only the K decoded reference frames and does not have to warp N complete frames for motion compensation. Rather, for each block or macroblock that is compensated using an affine motion parameter set, the translational motion vector and the affine motion model, which is indicated via the model index, are combined to obtain the displacement field for that image segment.

Figures 2 and 3 show an example for affine multi-frame prediction. The left hand frame (a) in Fig. 2 is the most recent decoded frame that would be the only frame to predict the right hand frame (b) in Fig. 2 in standard-based video compression. In this work, we additionally provide reference frames of which four are shown in Fig. 3. Hence, instead of just searching over the previous decoded frame (Fig. 2a), the block-based motion estimator can also search positions in the additional reference frames like the ones depicted in Fig. 3 and transmits the corresponding spatial displacement and frame selection parameter.

A. Video Syntax

In a well-designed video codec, the most efficient concepts should be combined in such a way that their utility can be adapted to the source signal without significant bit-rate overhead. More precisely, if the bit-rate overhead needed to signal an encoding strategy is larger than the bit-rate savings achieved, the strategy should be removed from the video coding syntax. This idea is incorporated into the design of many video codecs. Hence, the proposed video codec enables the utilization of variable block-size coding, long-term memory prediction and affine motion compensation in a rate-distortion efficient way. The use of the various motion compensation schemes can be signaled with very little overhead.

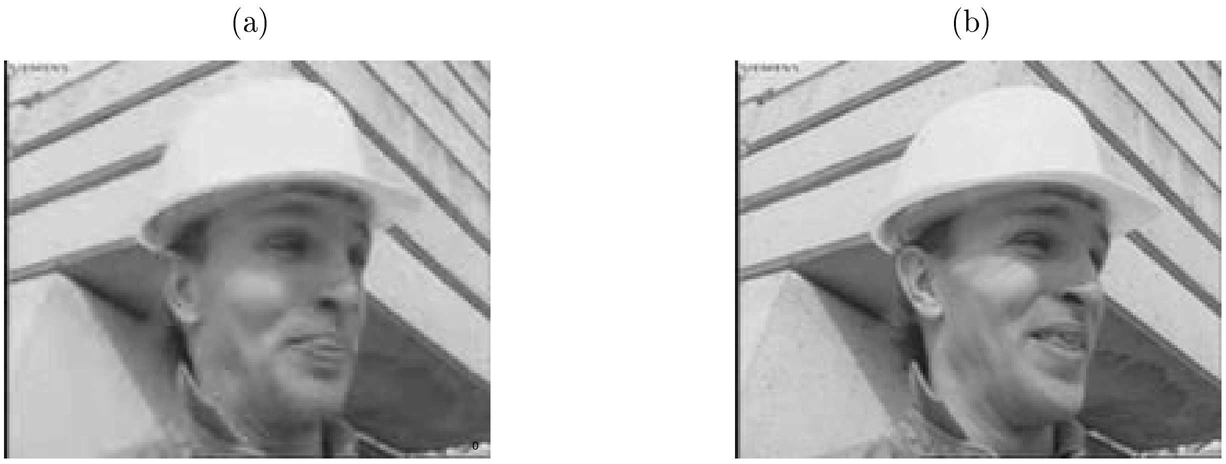

Fig. 2. Two frames from the QCIF test sequence *Foreman*, (a): previous decoded frame, (b): original frame.

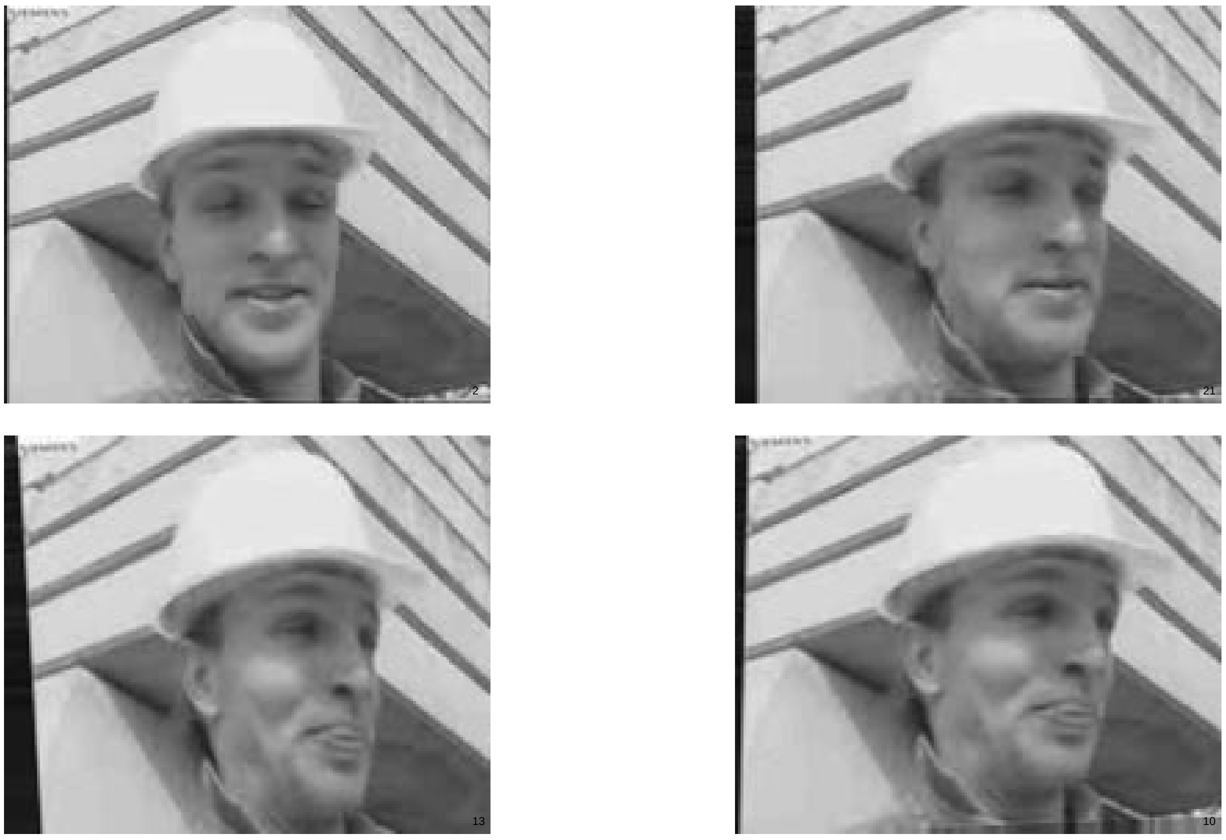

Fig. 3. Four additional reference frames. The upper left frame is a decoded frame that was transmitted 2 frame intervals before the previous decoded frame. The upper right frame is a warped version of the decoded frame that was transmitted 1 frame interval before the previous frame. The lower two frames are warped versions of the previous decoded frame.

The affine long-term memory motion-compensated predictor is integrated into an H.263+-based video coding scheme. Our motivation for that is: (*i*) the algorithm is well defined and state-of-the-art [2], (*ii*) the highly optimized test model of the H.263+ standard, TMN-10, can be used as a reference for comparison.

The parameters for the M^* chosen reference frames are transmitted in the header of each picture. First, their actual number using a variable length code is signaled. Then, the temporal reference of each reference frame is encoded followed by a bit indicating whether the frame is warped or not. If that bit indicates a warped frame, six affine motion parameters are transmitted. The motion model employed in our codec is an orthogonalized version of the well known affine model as will be described later. Uniform scalar quantization of the motion parameters followed by entropy coding is used. For more details on syntax we refer to [26]. The syntax specifying the actual number of reference frames allows the adaptation of the encoded bit-stream to the source signal on a frame basis without incurring much overhead. Hence, if affine motion is not efficient, one bit is enough to turn it off.

Motion estimation and compensation are performed using block-based multi-frame prediction mainly following the syntax and specifications of H.263+. The H.263+ syntax is modified in that changes are made to the inter-prediction modes INTER, INTER-4V, and SKIP. The INTER and SKIP mode are assigned one code word representing the picture reference parameter for the entire macroblock. The INTER-4V mode utilizes four picture reference parameters each associated with one of the four 8×8 motion vectors. The syntax mainly follows the proposed Annex for long-term memory prediction [9].

B. Affine Motion Model

In this work we use an affine motion model that describes the relationship between the motion of planar objects and the observable motion field in the image plane via a parametric expression. This model can describe motion such as translation, rotation, and zoom using six parameters $\boldsymbol{a} = (a_1, a_2, a_3, a_4, a_5, a_6)^T$. For estimation and transmission of the affine motion parameter sets, we adopt the orthogonalization approach in [21]. The orthogonalized affine model is used to code the displacement field $(m_x[\boldsymbol{a}, x, y], m_y[\boldsymbol{a}, x, y])^T$ and to transmit the affine motion parameters using uniform scalar quantization and variable length codes. In [21] a comparison was made to other approaches indicating the efficiency of the orthogonalized motion model. The motion model used in our investigation is given as

$$\begin{aligned} m_x[\boldsymbol{a}, x, y] &= \frac{w-1}{2}\left[a_1 c_1 + a_2 c_2 \left(x - \frac{w-1}{2}\right) + a_3 c_3 \left(y - \frac{h-1}{2}\right)\right], \\ m_y[\boldsymbol{a}, x, y] &= \frac{h-1}{2}\left[a_4 c_1 + a_5 c_2 \left(x - \frac{w-1}{2}\right) + a_6 c_3 \left(y - \frac{h-1}{2}\right)\right]. \end{aligned} \quad (1)$$

in which x and y are pixel locations in the image with $0 \leq x < w$ and $0 \leq y < h$ and w as well as h being image width and height. The coefficients c_1, c_2, and c_3 in (1) are given as

$$\begin{aligned} c_1 &= \frac{1}{\sqrt{w \cdot h}}, \\ c_2 &= \sqrt{\frac{12}{w \cdot h \cdot (w-1) \cdot (w+1)}}, \\ c_3 &= \sqrt{\frac{12}{w \cdot h \cdot (h-1) \cdot (h+1)}}. \end{aligned} \tag{2}$$

We quantize the affine coefficients a_i by

$$\tilde{a}_i = \frac{Q(\Delta \cdot a_i)}{\Delta} \quad \text{and} \quad \Delta = 2, \tag{3}$$

where $Q(\cdot)$ means rounding to the nearest integer value. The quantization levels of the affine coefficients $q_i = \Delta \cdot \tilde{a}_i$ are entropy coded and transmitted. We have found experimentally that similar coding results are obtained when varying the coarseness of the motion coefficient quantizer Δ in (3) from 2 to 10. Values of Δ outside this range, i.e., larger than 10 or smaller than 2, adversely affect coding performance.

III. Rate-Constrained Coder Control

In the previous Section, the video architecture and syntax are described. Ideally, the coder control should determine the coding parameters so as to achieve a rate-distortion efficient representation. This problem is compounded by the fact that typical video sequences contain widely varying content and motion, that can be more effectively quantized if different strategies are permitted to code different regions. For the affine motion coder, we additionally face the problem that the affine parameters must be assigned to large image segments to obtain a rate-distortion efficient motion representation. These large image segments usually can not be chosen so as to partition the image uniformly. Our solution to this problem is as follows:

A. Estimate N affine motion parameter sets between the current and the K previous frames.

B. Generate the multi-frame buffer which is composed of K past decoded frames and N warped frames that correspond to the N affine motion parameter sets.

C. Conduct multi-frame block-based hybrid video encoding on the $M = N + K$ reference frames.

D. Determine the number of affine motion parameter sets that are efficient in terms of rate-distortion performance.

In the following, we will describe steps A-D in detail.

A. Affine Motion Parameter Estimation

A natural camera-view scene may contain multiple independently moving objects in combination with camera motion and focal length change. Hence, region-based coding attempts to separate the effects via a scene segmentation and to code them accordingly. In this work, an explicit segmentation of the scene is avoided. Instead, the image is partitioned into blocks of fixed size and a fixed number of motion models N is estimated from decoded frames with respect to the current original frame.

For each cluster the affine motion coefficient estimation is performed in four steps.

1. Estimation of L translational motion vectors as initialization to the affine refinement.
2. Affine refinement of each of the L motion vectors using an image intensity gradient-based approach.
3. Concatenation of the initial translational and the affine refinement parameters.
4. Selection of one candidate among the L estimated affine motion models.

For the first step, block matching in the long-term memory buffer is performed in order to robustly deal with large displacements yielding L motion vectors. In the second step, the L translational motion vectors initialize an affine estimation routine which is based on image intensity gradients. The affine motion parameters are estimated by solving an over-determined set of linear equations so as to minimize mean squared error. In the third step, the resulting affine motion model is obtained by a weighted summation of the initial translational motion vector and the affine motion parameters. In the last step, the optimum in terms of mean squared error among the L considered candidates is chosen. In the following, the various steps are discussed in detail.

The initial motion vector estimation for a particular cluster comprising the pixels in the set $\mathcal{M}$ can be conducted in several ways. For instance, one translational motion vector which is the best match in terms of minimum mean squared error for all pixels in $\mathcal{M}$ with regard to each reference could be estimated. In that case, the number of considered candidates L would be equal to the number of decoded reference frames K. This approach that we call *cluster-based initialization* was proposed in [24], [25]. It provides flexibility in the choice of the cluster size and with that the number of clusters N. Hence, it will be used in Section IV to analyze the trade-off between rate-distortion performance and complexity that is inherent to the selection of the number of initial clusters.

However, the *cluster-based initialization* approach produces a computational burden that increases as the number of decoded reference frames K grows. On the other hand, we have to conduct translational motion estimation anyway for 16×16 blocks in H.263 and the long-term memory coder [8]. Hence, if we could re-use those motion vectors, we would drastically reduce the computational burden of the affine motion estimation. We call this approach the *macroblock-based initialization.* Therefore, an image partition is considered where the clusters are aligned with the macroblock boundaries. An example for such an initial partition is depicted in Fig. 4. Fig. 4 shows a QCIF picture from the sequence *Foreman* that is superimposed with 99 blocks of size 16×16 pixels. Each of these blocks is associated to the H.263 syntax unit of a macroblock for which either one or four motion vectors are transmitted in case that the macroblock is coded predictively. The $N = 20$ clusters are either blocks of size 32×32 pixels comprising 4 macroblocks, or blocks of size 32×48, 48×32, or 48×48 pixels.

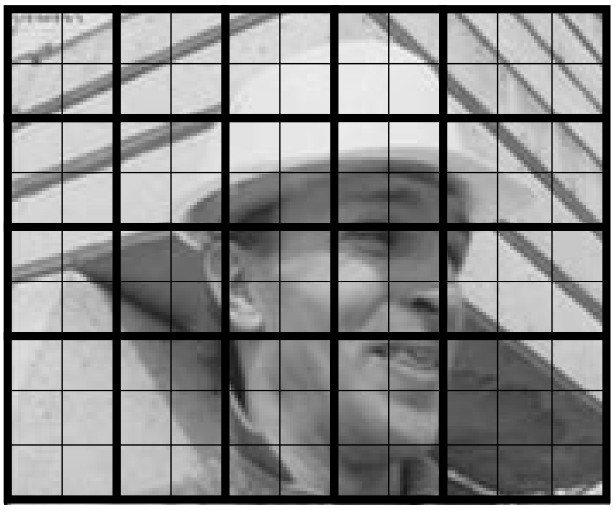
Fig. 4. Image partition of a QCIF frame of the sequence *Foreman* into 20 clusters.

If we utilize the motion vector of each macroblock as an initialization to the affine refinement step, we obtain either $L = 4, 6$ or 9 candidates. This number is independent from the number of decoded reference frames K. The motion estimation for the macroblocks proceeds as follows. To obtain the initial motion vector $\boldsymbol{m}^I = (m_x^I, m_y^I, m_t^I)^T$ which contains the spatial displacements m_x^I and m_y^I as well as the picture reference parameter m_t^I, a Lagrangian cost function is minimized which is given as

$$\boldsymbol{m}^I = \underset{\boldsymbol{m} \in \mathcal{S}}{\operatorname{argmin}} \{D_{DFD}(\boldsymbol{\mathcal{B}}, \boldsymbol{m}) + \lambda_{MOTION} \cdot R(\boldsymbol{m})\} \tag{4}$$

where the distortion $D_{DFD}(\boldsymbol{\mathcal{B}}, \boldsymbol{m})$ for the 16×16 block $\boldsymbol{\mathcal{B}}$ between the current frame $s[x, y, t]$ and the decoded reference frame $\tilde{s}[x, y, t - m_t]$ is computed as

$$D_{DFD}(\boldsymbol{\mathcal{B}}, \boldsymbol{m}) = \sum_{x,y \in \mathcal{B}} (s[x, y, t] - \tilde{s}[x - m_x, y - m_y, t - m_t])^2 . \tag{5}$$

$R(\boldsymbol{m})$ is the bit-rate associated with the motion vector. The minimization proceeds over the search space $\mathcal{S} = [-16 \ldots 16] \times [-16 \ldots 16] \times [0 \ldots K-1]$. First, the integer-pel motion vectors are determined that minimize the Lagrangian cost term in (4) for each of the K reference frames. Then, these K integer-pel accurate motion vectors are used as initialization of a half-pel refinement step which tests the 8 surrounding half-pel positions. Finally, the motion vector among the K candidates is determined as $\boldsymbol{m}^I$ which minimizes the Lagrangian cost term in (4). Following [1], the Lagrange multiplier is chosen as $\lambda_{MOTION} = 0.85 \cdot Q^2$, with Q being the DCT quantizer value, i.e., half the quantizer step size [2].

The initial translational motion vector $\boldsymbol{m}^I = (m_x^I, m_y^I, m_t^I)$ which is either obtained via the *cluster-based* or *macroblock-based initialization* is used to motion-compensate the past decoded frame $\tilde{s}[x, y, t - m_t]$ towards the current frame $s[x, y, t]$ as follows

$$\hat{s}[x, y, t] = \tilde{s}[x - m_x^I, y - m_y^I, t - m_t^I]. \tag{6}$$

This motion compensation is conducted for the pixels inside the considered cluster because those are used for the second step.

In the second step, we estimate an affine motion model against the translational motion-compensated signal $\hat{s}[x, y, t]$ for the pixels of the cluster collected in $\mathcal{M}$. The minimization criterion reads as follows

$$\boldsymbol{a}^R = \underset{\boldsymbol{a}}{\operatorname{argmin}} \sum_{x,y \in \mathcal{M}} (u[x, y, t, \boldsymbol{a}])^2 \tag{7}$$

with

$$u[x, y, t, \boldsymbol{a}] = s[x, y, t] - \hat{s}[x - m_x[\boldsymbol{a}, x, y], y - m_y[\boldsymbol{a}, x, y], t] \tag{8}$$

and $m_x[\boldsymbol{a}, x, y]$ as well as $m_y[\boldsymbol{a}, x, y]$ being given via (1).

Similar to [15], we expand $\hat{s}[x - m_x[\boldsymbol{a}, x, y], y - m_y[\boldsymbol{a}, x, y], t]$ into a first-order Taylor series around the spatial location (x, y) for small spatial displacements obtaining

$$\hat{s}[x - m_x[\boldsymbol{a}, x, y], y - m_y[\boldsymbol{a}, x, y], t] = \hat{s}[x, y, t] - \frac{\partial \hat{s}[x, y, t]}{\partial x} m_x[\boldsymbol{a}, x, y] - \frac{\partial \hat{s}[x, y, t]}{\partial y} m_y[\boldsymbol{a}, x, y]. \tag{9}$$

Hence, the error signal reads

$$u[x, y, t, \boldsymbol{a}] = s[x, y, t] - \hat{s}[x, y, t] + \frac{\partial \hat{s}[x, y, t]}{\partial x} m_x[\boldsymbol{a}, x, y] + \frac{\partial \hat{s}[x, y, t]}{\partial y} m_y[\boldsymbol{a}, x, y]. \tag{10}$$

Plugging (1) into (10) and rearrangement leads to the following linear equation with 6 unknowns

$$\begin{aligned} u[x, y, t, \boldsymbol{a}] = s[x, y, t] - \hat{s}[x, y, t] + \\ (g_x c_1, g_x c_2 x', g_x c_3 y', g_y c_1, g_y c_2 x', g_y c_3 y') \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{pmatrix} \end{aligned} \tag{11}$$

with the abbreviations

$$\begin{aligned} g_x &= \frac{w-1}{2} \frac{\partial \hat{s}[x, y, t]}{\partial x}, & g_y &= \frac{h-1}{2} \frac{\partial \hat{s}[x, y, t]}{\partial y}, \\ x' &= \left(x - \frac{w-1}{2}\right), & y' &= \left(y - \frac{h-1}{2}\right). \end{aligned} \tag{12}$$

Setting up this equation at each pixel position inside the cluster leads to an over-determined set of linear equations that is solved so as to minimize the average squared motion compensation error $u[x, y, t, \boldsymbol{a}]$. In this work, the pseudo inverse technique is used which is implemented via singular value decomposition. The inherent linearization holds for small displacements only. However, due to the translational initialization and the subsequent quantization of the affine motion coefficients it turns out that we do not need to iterate the affine motion parameter estimation in (11) several times. Experiments verify this statement, where we have varied the number of iterations without observing a significant difference.

The spatial intensity gradients are computed following [27], [28]. With $z \in \{x, y\}$ the spatial gradients are given as

$$\frac{\partial \hat{s}[x,y,t]}{\partial z} = \frac{1}{4}\sum_{i=0}^{1}\sum_{j=0}^{1} a_{ij}^{z} s[x+i, y+j, t] + b_{ij}^{z}\hat{s}[x+i, y+j, t], \tag{13}$$

With a_{ij}^z as well as b_{ij}^z being the element on the jth row and ith column of the matrices

$$\begin{aligned} \boldsymbol{A}^x = \boldsymbol{B}^x &= \begin{pmatrix} 1 & -1 \\ 1 & -1 \end{pmatrix} \\ \boldsymbol{A}^y = \boldsymbol{B}^y &= \begin{pmatrix} 1 & 1 \\ -1 & -1 \end{pmatrix} \end{aligned} \tag{14}$$

The estimates provide the gradient of the point in-between the four samples and between the pre-compensated and the current image [28]. Since the spatial gradients are computed between the pixel positions we also compute the frame difference $s[x, y, t] - \hat{s}[x, y, t]$ using the summation on the right hand side of (13) with $z = t$ and

$$\boldsymbol{A}^t = -\boldsymbol{B}^t = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}. \tag{15}$$

In the third step, the affine motion parameter for motion compensation between the reference frame $\tilde{s}[x, y, t - m_t^I]$ and the current frame $s[x, y, t]$ is obtained via concatenating the initial translational motion vector $\boldsymbol{m}^I$ and the estimated affine refinement model $\boldsymbol{a}^R$ yielding

$$\begin{aligned} a_1 &= \frac{2m_x^I}{c_1(w-1)} + a_1^R \\ a_2 &= a_2^R \\ a_3 &= a_3^R \\ a_4 &= \frac{2m_y^I}{c_1(h-1)} + a_4^R \\ a_5 &= a_5^R \\ a_6 &= a_6^R \end{aligned} \tag{16}$$

The initial translational block matching and the affine refinement procedure are repeated for each of the L candidates. In the last step, the affine motion model is chosen that minimizes the mean squared error measured over the pixels in the cluster $\mathcal{M}$.

B. Reference Picture Warping

For each of the N estimated motion models $\boldsymbol{a}_i$, $i = 1 \ldots N$, we select the reference frame corresponding to m_t^I and warp it towards the current frame which is to be coded. The reference picture warping is conducted using the motion field that is computed via (1) for each motion model $\boldsymbol{a}_i$ for the complete frame. Non-integer displacements are computed using cubic spline interpolation [29] which turns out to be more efficient than bi-linear interpolation as the motion model becomes more complex [30]. We therefore obtain N new reference frames that can be used for block-based prediction of the current frame as illustrated in Fig. 1.

C. Rate-Constrained Multi-Frame Hybrid Video Encoding

At this point it is important to note that the multi-frame buffer is filled with the K most recent frames and N warped frames yielding M reference frames. In order to produce the motion-compensated frame, we conduct multi-frame block-based motion compensation similar to H.263. That is, half-pel accurate motion vectors $\boldsymbol{m} = (m_x, m_y, m_t)^T$ are applied to compensate blocks of size 16×16 pixels referencing one of the $M = K + N$ reference frames. Again, block-based motion estimation is conducted to obtain the motion vectors by minimizing (4) as it was done when searching decoded frames to initialize affine motion estimation. In case the *macroblock-based initialization* is used, we can re-use the motion vectors. Otherwise, motion estimation over the K decoded frames has to be conducted as described for the *macroblock-based initialization*. When searching the warped reference frames, we actually have to search only a range of $[-2 \ldots 2] \times [-2 \ldots 2]$ spatially displaced pixels. The small search range is justified by the fact that the warped frames are already motion-compensated and experiments show that only a very small percentage of motion vectors is found outside this range.

Given the motion vectors, the Lagrangian costs for each macroblock mode are determined. This is a pre-computation step for the algorithm that determines the number of efficient reference frames and with that affine motion models. The Lagrangian cost function [5] reads as

$$D_{REC}(\mathcal{B}, h, \boldsymbol{m}, c) + \lambda_{MODE} \cdot R_{REC}(\mathcal{B}, h, \boldsymbol{m}, c). \tag{17}$$

Here, the distortion after reconstruction D_{REC} measured as the sum of squared differences between pixels of the block $\mathcal{B}$ in the original and reconstructed frame is weighted against bit-rate R_{REC} using the Lagrange multiplier λ_{MODE}. R_{REC} is the bit-rate that is needed to transmit a particular mode, including the macroblock header h, motion information $\boldsymbol{m}$, and DCT coefficients c. Following [1], the Lagrange multiplier for the mode decision is chosen as $\lambda_{MODE} = \lambda_{MOTION} = 0.85 \cdot Q^2$. The

Lagrangian cost functions are determined for the macroblock modes INTER, SKIP, and INTRA [2] and the values are stored in an array to provide fast access.

D. Determination of the Number of Reference Frames

As mentioned before, there is still an open problem about the efficient combination of motion vectors, macroblock modes and reference frames. Because of the inter-dependency of the various parameters a locally optimal solution is searched using the pre-computed rate-distortion costs. The greedy optimization algorithm reads as follows

1. Sort the reference frames according to the frequency of their selection.
2. Starting with the least popular reference frame, test the utility of each reference frame by
   (a) Computing its best replacement among the more popular frames in terms of rate-distortion costs block by block.
   (b) If the costs for transmitting the reference frame parameters exceed the cost of using the replacements for this frame, remove the frame, otherwise keep it.

The first step is conducted because of the use of variable length codes to index the reference frames. The chosen reference frame with associated warping parameters are transmitted in the header of each picture. The order of their transmission provides the model index that is used to specify a particular reference frame on the block basis. This model index is entropy-coded.

In the second step, the utility of each reference frame is tested by evaluating the rate-distortion improvement obtained by removing this reference frame. For those blocks that reference the removed frame, the best replacements in terms of Lagrangian costs among the more popular reference frames are selected. If no rate-distortion improvement is observed, the frame is kept in the reference buffer and the procedure is repeated for the next reference frame.

After having determined the number of efficient frames M^* in the multiple reference frame buffer, the rate-distortion costs of the INTER-4V macroblock mode are considered and the selected parameters are encoded. Up to this point, the INTER-4V mode has been intentionally left out of the encoding because of the associated complexity to determine the mode costs.

IV. Experiments

Within the framework of the multi-frame affine motion coder there are various free parameters that can be adjusted. In this Section, we give empirical justifications for important parameter choices made. We concentrate on relevant aspects that have the largest impact on the trade-off between rate-distortion performance and computational complexity. Regarding the affine motion coder we will elaborate on the important question about the number of initial motion clusters N. This parameter is very critical since the number of warped reference pictures is directly affected by N.

Then, the combination of long-term memory prediction with affine motion compensation is analyzed. We will especially look at the relative increase in efficiency when comparing long-term memory prediction to affine motion coding in case of warping only the prior picture. Finally, the gains when combining affine and long-term memory prediction are investigated.

The experiments in this work were conducted using the QCIF test sequences and conditions in Tab. I. The first four sequences contain a large amount of motion including a moving camera and focal length changes. The last four sequences are low motion sequences with a fixed camera. This set of ITU-T test sequences was chosen so as to cover a broad range of possible scenes that might occur in applications such as video conferencing or video streaming.

Sequence Name	Abbre-viation	Number of Frames	Frame Skip	Global Motion
Foreman	fm	400	2	Yes
Mobile & Calendar	mc	300	2	Yes
Stefan	st	300	2	Yes
Tempete	te	260	1	Yes
Container Ship	cs	300	2	No
Mother & Daughter	md	300	2	No
News	nw	300	2	No
Silent	si	300	1	No

TABLE I

Set of test sequences.

For all experiments, bit-streams are generated that are decodable producing the same PSNR values at encoder and decoder. The first Intra-coded frame is identical for all cases considered.

A. The Affine Motion Coder

In this Section we investigate the parameter setting for the affine coder. For that, we restrict the warping to exclusively reference the prior decoded picture. As shown later, the results for this case also propagate to a setting where the affine motion coder is combined with long-term memory prediction.

The first question to clarify concerns the number of initial motion clusters. The translational motion vector estimation is conducted using the *cluster-based initialization* as described in Section III-A. We have initialized the coder with $N = 1, 2, 4, 8, 16, 32, 64$, and 99 clusters. The partition into the N clusters was achieved so as to obtain equal size blocks and each of the blocks being as close as possible to a square. The translational motion vectors serve as an initialization to the affine refinement step described in Section III-A. The estimated affine motion models are used to warp the previous decoded frame N times as explained in Section III-B. Block-based multi-frame motion estimation and determination of the number of efficient motion models is conducted as described in Sections III-C and III-D.

Figure 5 shows the average bit-rate savings for the set of test sequences in Tab. I. The average bit-rate savings are measured for each sequence at fixed PSNR values of 32, 34 and 36 dB. For that, rate-distortion curves are generated by varying the DCT quantizer and the Lagrange parameter accordingly. The bit-rate corresponds to the overall bit-rate that has to be transmitted to reconstruct each sequence at the decoder and distortion is computed as average PSNR over all frames. The intermediate points of the rate-distortion curves are interpolated and the bit-rate that corresponds to a given PSNR value is obtained. The curves in Fig. 5 are obtained via computing the mean of the bit-rate savings for each sequence. This procedure is conducted for all settings. We are aware that the percentage in bit-rate

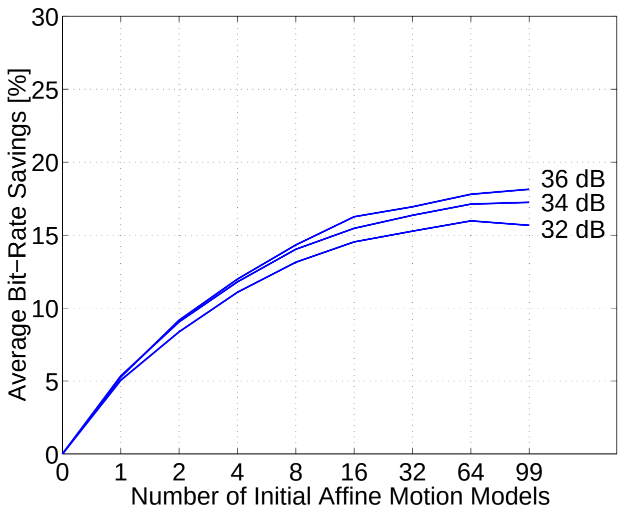

Fig. 5. Average bit-rate savings versus number of initial motion models for the test sequences in Tab. I.

savings corresponds to different absolute bit-rate values for the various sequences. Hence, later we will also show rate-distortion curves. Nevertheless, computing bit-rate savings might provide a meaningful measure, for example, for video content providers who want to guarantee a certain quality of the reconstructed sequences.

The average bit-rate savings are very similar for the three different levels of reproduction quality. The number of initial affine motion models has a significant impact on resulting rate-distortion performance. The increase in bit-rate savings tends to a saturation for a large number of motion models, i.e., more than 32 motion models, reaching the value of 17.8 % for our set of test sequences for the reproduction quality of 34 dB PSNR.

This can be explained when investigating the average number of motion models that were transmitted as shown in Fig. 6. The average number of motion models are generated with a similar method as the average bit-rate savings for a given PSNR value. The average number of motion models increases

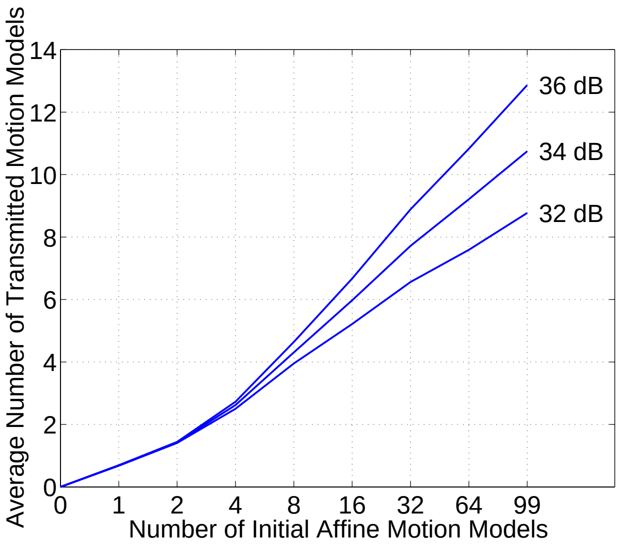

Fig. 6. Average number of transmitted motion models versus number of initial motion models for the test sequences in Tab. I.

with increasing average PSNR as well as an increased number of initial affine motion models. This is because the size of the measurement window becomes smaller as the number of initial affine motion models increases and the motion models are more accurate inside the measurement window. Hence, the coder chooses to transmit more motion models. For very small numbers of initial affine motion models, a large percentage of the possible number of motion models is chosen. However, as the number of initial motion models is increased, a decreasing percentage of motion models is transmitted.

Figure 7 shows the average bit-rate savings at 34 dB PSNR for the set of test sequences where the result for each sequence is shown using dashed lines. The abbreviations fm, mc, st, te, cs, md, nw, and si correspond to those in Tab. I. The solid line depicts the average bit-rate savings for the 8

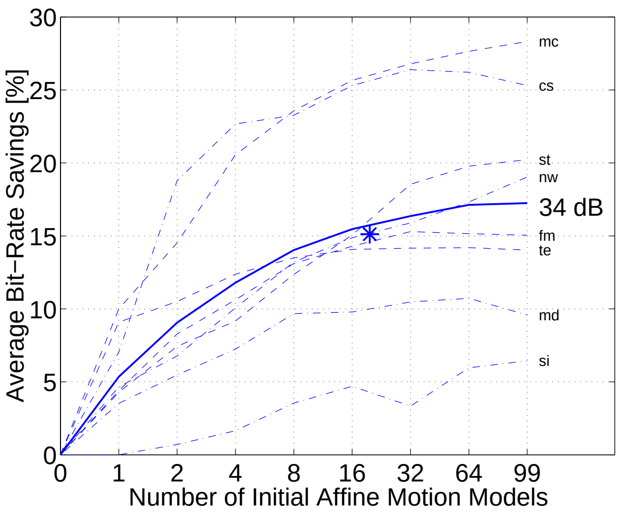

Fig. 7. Average bit-rate savings at 34 dB PSNR versus number of initial motion models for the test sequences in Tab. I.

test sequences at equal PSNR of 34 dB. The results differ quite significantly among the sequences in our test set. On one hand, for the sequence *Silent*, only bit-rate saving of 5% can be obtained. On the other hand, sequences like *Mobile & Calendar* and *Container Ship* show substantial gains of more than 25 % in bit-rate savings. We have observed that we obtain especially good results for sequences with large moving areas containing active texture as it is the case for *Container Ship* and *Mobile & Calendar*. Also camera motion can be well represented and thus efficiently coded with our scheme. We conjecture that two effects mainly contribute to the improved rate-distortion performance: (*i*) rate-distortion efficient MCP for motions such as rotation and zoom which are not captured well by a translational motion model and (*ii*) increased motion accuracy in conjunction with cubic spline interpolation.

In Fig. 7, the asterisk shows the average result for the *macroblock-based initialization* of the affine estimation (see Section III-A). Please recall that for all experiments that were described so far, we used the *cluster-based initialization* for the translational motion vector estimation to have a simple means for varying the number of initial clusters. For that, we employ the segmentation in Fig. 4 resulting in $N = 20$ clusters. The bit-rate saving of 15.1 % is very close to the results for the *cluster-based initialization*. However, the complexity is drastically reduced. Typical run-time numbers for the *macroblock-based initialization* are as follows. The complete affine motion coder runs at 6.5 seconds per QCIF frame on a 300 MHz Pentium PC. These 6.5 seconds are split into 0.5 seconds for translational motion estimation for 16×16 macroblocks, 1 second for affine motion estimation, and the warping also takes 1 second. The precomputation of the costs for the INTER, SKIP, and INTRA mode takes 2 seconds, and the remaining steps use 2 seconds. As a comparison, our H.263 coder which has a similar degree of run-time optimization uses 2 seconds per QCIF frame. Finally, let us depict rate-distortion curves for this approach.

The rate-distortion curves for the affine motion coder are shown in comparison to TMN-10, the state-of-the-art H.263 test model which is also operated using Lagrangian optimization methods [7], [6]. The following abbreviations will be used to indicate the two cases:

- **TMN-10:** The H.263 test model using Annexes D, F, I, J, and T.
- **MRPW:** As TMN-10, but motion compensation is extended to referencing warped frames corresponding to 20 affine motion models.

Figure 8 shows the rate-distortion curves for four test sequences from our set in Tab. I. The PSNR gains vary for the different test sequences and tend to be larger as the bit-rate increases. In contrast, the bit-rate savings in % are more or less constant over the entire range of bit-rates that was tested. Typically, we obtain a PSNR gain of 1 dB compared to TMN-10. The PSNR gains are up to 2.3 dB for the sequence *Mobile & Calendar*.

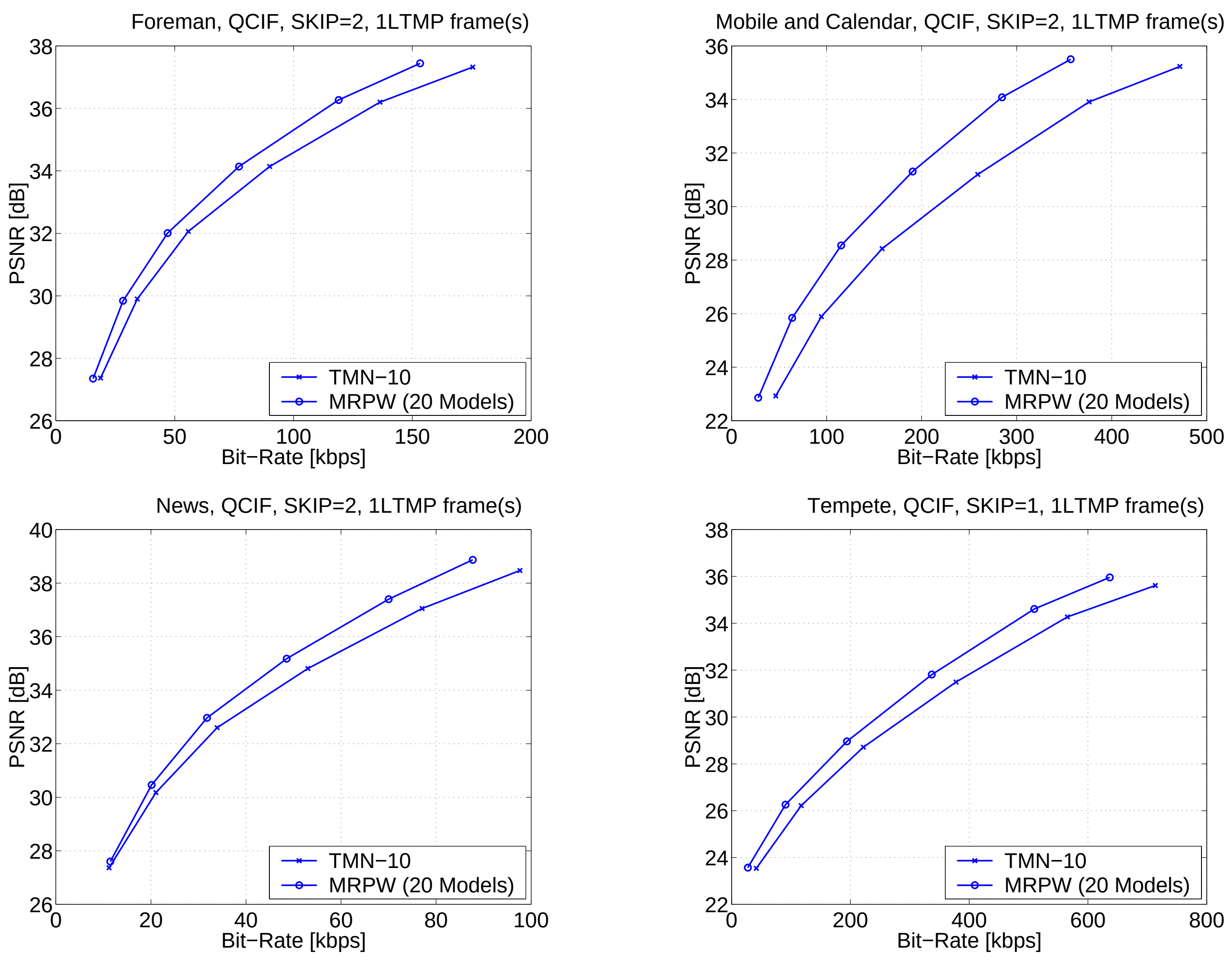

Fig. 8. PSNR vs. overall bit-rate for the QCIF sequences *Foreman* (top left), *Mobile & Calendar* (top right), *News* (bottom left), and *Tempete* (bottom right).

We have also conducted experiments with the corresponding versions of our test sequences in CIF resolution. In that case, similar performance measures were found.

B. Combination of Affine and Long-Term Memory Motion Compensation

In the previous Section, we have shown that affine motion compensation provides significant bit-rate savings. However, we also saw that increasing the number of motion models beyond 32 does not provide substantial benefits. Rather, the complexity of the proposed scheme would be drastically increased if we would use such large numbers. Hence, we arrived at a design that uses 20 initial clusters providing an average bit-rate saving of 15.1 %.

In contrast to the affine motion coder where warped versions of the prior decoded frame are employed, the long-term memory prediction coder references past decoded frames for motion compensation. However, aside from the different origin of the various reference frames, the syntax for both codecs is very similar. Figure 9 shows the average bit-rate savings at 34 dB PSNR for the set of test sequences that are achieved with the long-term memory prediction coder as described in [8]. The abbreviations fm, mc, st, te, cs, md, nw, and si correspond to those in Tab. I. The methodology to obtain the average bit-rate savings is the same as for the affine motion codec in the previous Section. The average bit-rate savings achieved with the long-term memory codec are in a similar range as for the affine motion codec.

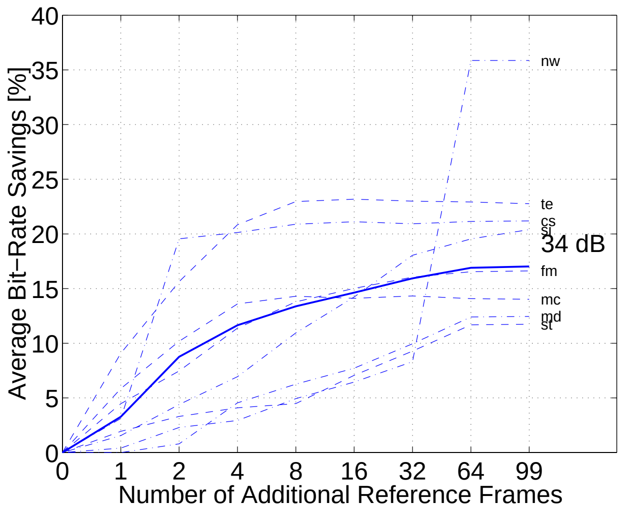

Fig. 9. Average bit-rate savings at 34 dB PSNR versus number of additionally buffered reference frames for the set of test sequences in Tab. I.

The sequence *News* (nw) shows a strong increase in bit-rate savings for 64 or 99 additional reference frames compared to 32 additional frames. This MPEG-4 test sequence is an artificial sequence where in the background two distinct sequences of dancers are displayed. These sequences, however, are repeated every 2.5 seconds corresponding to 25 frames in the long-term memory buffer. Hence, our long-term memory coder obtains extremely large gains for memory $K \geq 50$ when comparing to the TMN-10 result, since in that case the old dancer sequence is still available in the long-term memory buffer. The bold line in Fig. 9 shows the average bit-rate savings when excluding the sequence *News* from the set of test sequences.

We achieve an average bit-rate reduction of 17 % when utilizing 99 additional reference frames in our long-term memory coder. The bit-rate savings saturate as we further increase the number of reference frames. Already when utilizing 9 additional reference frames, i.e., using $K = 10$ reference frames overall, we get 13.8 % average bit-rate savings. Hence, we will use 10 decoded frames when combining long-term memory prediction and affine motion compensation. Please note that if we again insert the *News* sequence into our set of test sequences, we obtain 12.5 % bit-rate savings when using $K = 10$ frames.

In Fig. 10, the result when combining the affine motion coder and long-term memory prediction is depicted. This plot shows average bit-rate savings at 34 dB PSNR versus the number of initial affine motion clusters for the set of test sequences in Tab. I. Two cases are shown: (i) affine warping using

$K = 1$ reference frame (lower solid curve) and (*ii*) affine warping using $K = 10$ reference frames (upper solid curve). For the case $K = 1$, the setting of the coder has been employed again that was used for the curve depicting the average bit-rate savings at 34 dB in Fig. 5. To obtain the result for the case $K = 10$, we again run the *cluster-based initialization* with $N = 1, 2, 4, 8, 16, 32, 64$, and 99 clusters. For the *cluster-based initialization* of the affine motion estimation, $L = K = 10$ initial translational motion vectors are utilized each corresponding to the best match on one of the K reference frames (see Section III-A). Please note that the number of maximally used reference frames is $N + K$. Interestingly, the average bit-rate savings obtained by the affine motion and the long-term memory prediction coder are almost additive when being combined using multi-frame affine MCP.

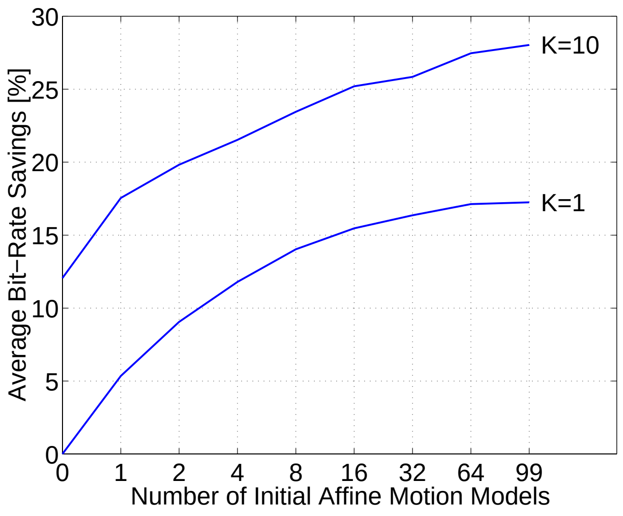

Fig. 10. Average bit-rate savings at 34 dB PSNR versus number of initial affine motion models for the set of test sequences in Tab. I. Two cases are shown: (*i*) affine warping using $K = 1$ reference frames (lower solid curve) and (*ii*) affine warping using $K = 10$ reference frames (upper solid curve).

Figure 11 shows the bit-rate savings for each of our test sequences in Tab. I when employing $K = 10$ decoded reference frames versus the number of initial affine motion models N using dashed lines. The solid line for $K = 10$ is repeated from Fig. 5. The bit-rate savings are more than 35 % for the sequences *Container Ship* and *Mobile & Calendar* when using 32 or more initial affine motion models. Interestingly, when using $K = 10$ reference frames and 16 or more initial affine motion models we are never below 15 % bit-rate savings. Moreover, for some sequences the gain obtained by the combined coder is larger as the added gains of the two separate coders. For example, the long-term memory prediction gain for *Mother & Daughter* is 7 % for $K = 10$ frames. The gain obtained for the affine motion coder is 10 % when using 32 initial affine models. However, the combined coder achieves 23 % bit-rate savings for the sequence *Mother & Daughter*.

In Fig. 5, the asterisk shows the result for the case if *macroblock-based initialization*. For that, we employ the segmentation in Fig. 4 resulting in $N = 20$ clusters. The initial motion vectors for the affine motion estimation are those best matches found for the macroblocks in each cluster when searching

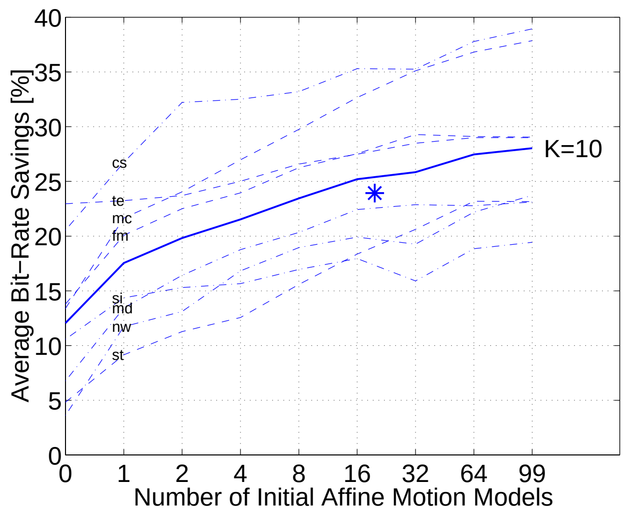

Fig. 11. Average bit-rate savings at 34 dB PSNR versus number of initial affine motion models for the set of test sequences in Tab. I.

$K = 10$ decoded reference frames. We obtain an average bit-rate saving of 24 % over our set of 8 test sequences in Tab. I. Finally, let us depict rate-distortion curves for this approach.

Figure 12 shows rate-distortion curves that are produced by the following three codecs

- **TMN-10:** The H.263 test model using Annexes D, F, I, J, and T [6].
- **LTMP:** As TMN-10, but motion compensation is extended to long-term memory prediction with $K = 10$ decoded reference frames according to [8].
- **MRPW+LTMP:** As TMN-10, but motion compensation is extended to combined affine and long-term memory prediction. The size of the long-term memory is selected as $K = 10$ frames. The number of estimated affine motion models is $N = 20$.

Long-term memory MCP with 10 frames and without affine warping is always better than TMN-10 as already demonstrated in [8]. Moreover, long-term memory MCP in combination with affine warping is always better than the case without affine warping. Bit-rate savings up to 35 % can be achieved that correspond to PSNR gains of 3 dB. For some sequences long-term memory prediction provides the most gain (*Tempete*) while for other sequences the affine motion coder is more important (*Mother & Daughter*).

V. Conclusions

The extension of long-term memory prediction by affine reference picture warping yields a superior video coding scheme in terms of rate-distortion performance. When warping the prior decoded frame, average bit-rate savings of 15.1 % against TMN-10 are reported for the case that 20 initial affine motion models are used. For the measurements, reconstruction PSNR is equal to 34 dB for all cases considered. These average bit-rate savings are measured over a set of 8 test sequences that represent

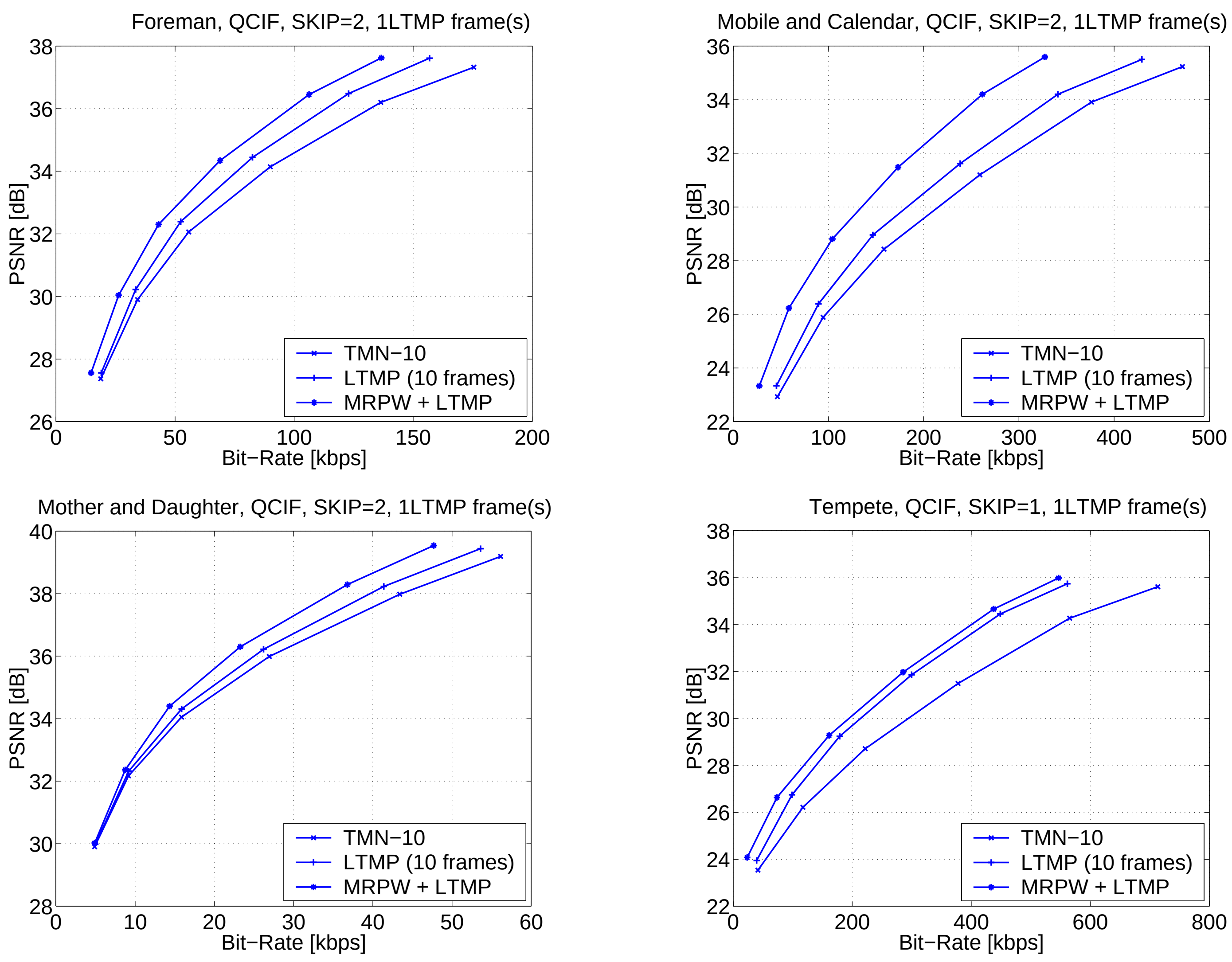

Fig. 12. PSNR vs. overall bit-rate for the QCIF sequences *Foreman* (top left), *Mobile & Calendar* (top right), *Mother & Daughter* (bottom left), and *Tempete* (bottom right).

a large variety of video content. Within the test set, these gains vary from 5 % to 25 %.

Long-term memory prediction has been already demonstrated as an efficient means to compress motion video. The efficiency in terms of rate-distortion performance is comparable to that of the affine coder. The combination of the two approaches yields almost additive average gains. When employing 20 initial affine motion models and 10 reference frames, we obtain 24 % in average bit-rate savings against TMN-10. The minimum bit-rate saving inside our test set is 15 % while maximum saving is reported to be up to 35 %. These bit-rate savings correspond to gains in PSNR between 0.8 and 3 dB. We have also found cases where the combination of affine and long-term memory prediction provide more than additive gains. For subjective evaluations, bit-streams and a decoder can be down-loaded via anonymous ftp from `ftp.lnt.de/pub/wiegand/MRPW`.

References

[1] G. J. Sullivan and T. Wiegand, “Rate-Distortion Optimization for Video Compression”, *IEEE Signal Processing Magazine*, vol. 15, no. 6, pp. 74–90, Nov. 1998.

[2] ITU-T Recommendation H.263 Version 2 (H.263+), “Video Coding for Low Bitrate Communication”, Jan. 1998.

[3] G. J. Sullivan and R. L. Baker, “Rate-Distortion Optimized Motion Compensation for Video Compression Using Fixed or Variable Size Blocks”, in *Proc. GLOBECOM’91*, 1991, pp. 85–90.

[4] B. Girod, “Rate-Constrained Motion Estimation”, in *Proceedings of the SPIE Conference on Visual Communications and Image Processing*, Chicago, USA, Sept. 1994, pp. 1026–1034.

[5] T. Wiegand, M. Lightstone, D. Mukherjee, T. G. Campbell, and S. K. Mitra, “Rate-Distortion Optimized Mode Selection for Very Low Bit Rate Video Coding and the Emerging H.263 Standard”, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 2, pp. 182–190, Apr. 1996.

[6] ITU-T/SG16/Q15-D-65, “Video Codec Test Model, Near Term, Version 10 (TMN-10), Draft 1”, Download via anonymous ftp to: standard.pictel.com/video-site/9804_Tam/q15d65d1.doc, Apr. 1998.

[7] ITU-T/SG16/Q15-D-13, T. Wiegand and B. Andrews, “An Improved H.263-Codec Using Rate-Distortion Optimization”, Download via anonymous ftp to: standard.pictel.com/video-site/9804_Tam/q15d13.doc, Apr. 1998.

[8] T. Wiegand, X. Zhang, and B. Girod, “Long-Term Memory Motion-Compensated Prediction”, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no. 1, pp. 70–84, Feb. 1999.

[9] ITU-T/SG16/Q15-G-18, T. Wiegand, N. Färber, B. Girod, and B. Andrews, “Proposed Draft for Annex U on Enhanced Reference Picture Selection”, Download via anonymous ftp to: standard.pictel.com/video-site/9902_Mon/q15g18.doc, Feb. 1999.

[10] R. Y. Tsai and T. S. Huang, “Estimating Three-Dimensional Motion Parameters of a Rigid Planar Patch”, *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 29, no. 6, pp. 1147–1152, Dec. 1981.

[11] M. Hötter and R. Thoma, “Image Segmentation Based on Object Oriented Mapping Parameter Estimation”, *Signal Processing: Image Communication*, vol. 16, pp. 315–334, Oct. 1988.

[12] N. Diehl, “Object-Oriented Motion Estimation and Segmentation in Image Sequences”, *Signal Processing: Image Communication*, vol. 3, pp. 23–56, Jan. 1991.

[13] H. Sanson, “Motion Affine Models Identification and Application to Television Image Sequences”, in *Proceedings of the SPIE Conference on Visual Communications and Image Processing*, 1991, vol. 1605, pp. 570–581.

[14] Y. Yokoyama, Y. Miyamoto, and M. Ohta, “Very Low Bit Rate Video Coding Using Arbitrarily Shaped Region-Based Motion Compensation”, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, no. 6, pp. 500–507, Dec. 1995.

[15] C. K. Cheong, K. Aizawa, T. Saito, M. Kaneko, and H. Harashima, “Structural Motion Segmentation for Compact Image Sequence Representation”, in *Proceedings of the SPIE Conference on Visual Communications and Image Processing*, Orlando, FL, USA, Mar. 1996, pp. 1152–1163.

[16] E. Francois, J.-F. Vial, and B. Chupeau, “Coding Algorithm with Region-Based Motion Compensation”, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 1, pp. 97–108, Feb. 1987.

[17] S.-C. Han and J. W. Woods, “Adaptive Coding of Moving Objects for Very Low Bit Rates”, *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 1, pp. 56–70, Jan. 1998.

[18] H. Li and R. Forchheimer, “A Transform Block-Based Motion Compensation Technique”, *IEEE Transactions on Communications*, vol. 43, no. 2, pp. 1673–1676, Feb. 1995.

[19] H. Jozawa, K. Kamikura, A. Sagata, H. Kotera, and H. Watanabe, “Two-Stage Motion Compensation Using Adaptive Global MC and Local Affine MC”, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 1, pp. 75–85, Feb. 1997.

[20] K. Zhang, M. Bober, and J. Kittler, “Image Sequence Coding Using Multiple-Level Segmentation and Affine Motion Estimation”, *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 9, pp. 1704–1713, Dec. 1997.

[21] M. Karczewicz, J. Niewęgłowski, and P. Haavisto, “Video Coding Using Motion Compensation with Polynomial Motion Vector Fields”, *Signal Processing: Image Communication*, vol. 10, pp. 63–91, 1997.

[22] T. Wiegand, E. Steinbach, A. Stensrud, and B. Girod, “Multiple Reference Picture Coding using Polynomial Motion

Models", in *Proceedings of the SPIE Conference on Visual Communications and Image Processing*, San Jose, USA, Feb. 1998, pp. 134–145.

[23] D. Lauzon and E. Dubois, "Representation and Estimation of Motion Using a Dictionary of Models", in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 1998, pp. 2585–2588.

[24] E. Steinbach, T. Wiegand, and B. Girod, "Using Multiple Global Motion Models for Improved Block-Based Video Coding", in *Proceedings of the IEEE International Conference on Image Processing*, Kobe, Japan, Oct. 1999, vol. 2, pp. 56–60.

[25] T. Wiegand, E. Steinbach, and B.Girod, "Long-Term Memory Prediction Using Affine Motion Compensation", in *Proceedings of the IEEE International Conference on Image Processing*, Kobe, Japan, Oct. 1999, vol. 1, pp. 51–54.

[26] ITU-T/SG16/Q15-G-21, T. Wiegand, E. Steinbach, B. Girod, and B. Andrews, "Video Coding Using Long-Term Memory and Affine Motion Compensation", Download via anonymous ftp to: standard.pictel.com/ video-site/9902_Mon/q15g21.doc, Feb. 1999.

[27] B. K. P. Horn and B. G. Schunk, "Determining Optical Flow", *Artificial Intelligence*, vol. 17, pp. 185–203, 1981.

[28] B. K. P. Horn, *Robot Vision*, The MIT Press, McGraw-Hill Book Company, USA, 1986.

[29] M. Unser, "Splines: A Perfect Fit for Signal and Image Processing", *IEEE Signal Processing Magazine*, vol. 16, no. 6, pp. 22–38, Nov. 1999.

[30] J.-L. Dugelay and H. Sanson, "Differential Methods for the Identification of 2D and 3D Motion Models in Image Sequences", *Signal Processing: Image Communication*, vol. 7, pp. 105–127, 1995.