Affine Multi-Frame Motion-Compensated Prediction

T. Wiegand[1], E. G. Steinbach[2], and B. Girod[2]

Image Processing Department
Heinrich Hertz Institute
Berlin, Germany
wiegand@HHI.de

Information Systems Laboratory
Stanford University
Stanford, CA
[steinb,bgirod]@stanford.de

Abstract

Affine motion compensation is combined with long-term memory motion-compensated prediction. The idea is to determine several affine motion parameter vectors on sub-areas of the image. Then, for each affine motion parameter vector, a complete reference frame is warped and inserted into the multi-frame buffer. Given the multi-frame buffer of decoded frames and affine warped versions thereof, block-based translational motion-compensated prediction and Lagrangian coder control are utilized. The affine motion parameters are transmitted as side information requiring additional bit-rate. Hence, the utility of each reference frame and with that each affine motion parameter vector is tested for its rate-distortion efficiency. The combination of affine and long-term memory motion-compensated prediction provides an highly efficient video compression scheme in terms of rate-distortion performance. The two incorporated multi-frame concepts complement each other well providing almost additive rate-distortion gains. When warping the prior decoded frame, average bit-rate savings of 15 % against TMN-10, the test model of ITU-T Recommendation H.263+, are reported for the case that 20 warped reference pictures are used. When employing 20 warped reference pictures and 10 decoded reference frames, average bit-rate savings of 24 % can be obtained for a set of 8 test sequences. These bit-rate savings correspond to gains in PSNR between 0.8 and 3 dB. For some cases, the combination of affine and long-term memory MCP provides more than additive gains.

1 Introduction

The most successful class of today's video compression schemes are called hybrid codecs. 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 [2]. 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 [2, 3, 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 [4].

Long-term MCP [5, 6] 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 MCP is performed using motion vectors that consist of a spatial displacement and a picture reference to address a block in the multi-frame buffer. Rate-constrained motion estimation is employed to control the bit-rate of the motion data. The ITU-T/SG16/Q15 group has decided to adopt this feature as an Annex to the H.263 standard [7].

While long-term memory MCP extends the motion model to exploit long-term dependencies in the video sequence, the motion model remains translational. But, 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 as is the case when employing long-term memory MCP, this effect is further enhanced since more sophisticated motion is likely to occur. Hence, the efficiency of coding the motion information is often increased by enhancing the motion model.

In an early work, Tsai and Huang derive a parametric motion model that relates the motion of planar objects in the scene to the observable motion field in the image plane for a perspective projection model [8]. The eight parameters of this model are estimated using corresponding points [8]. A problem that very often occurs with the eight parameter model is that some parameters appear in the denominator of the parametric expression which adversely affects the parameter estimation procedure due to numerical problems. In [9], Hötter and Thoma approximate the planar object motion using a two-dimensional quadratic model of twelve parameters. The parameters are estimated using spatial and temporal intensity gradients which drastically improves the parameter estimates in the presence of noise.

In case the objects in the scene or the considered parts of the objects do not show large depth

variations with respect to the image plane, the simpler camera model of parallel projection can be applied. Popular motion models for parallel projection are the affine and bilinear motion model. Various researchers have utilized affine and bilinear motion models for *object-based* or *region-based* coding of image sequences [10, 11, 12, 13, 14, 15]. The motion parameters 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. Hence, other researchers have used affine or bilinear motion models in conjunction with a *block-based* approach to reduce the bit-rate for transmitting the image segmentation [16, 17]. 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 [18]. KARCZEWICZ *et al.* report in [18] that the use of the twelve parameter motion model 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.

Within the MPEG-4 standardization group, a technique called *Sprites* has been considered [19, 20, 21]. Sprites can exploit long-term statistical dependencies similar to background memory techniques [22, 23, 24, 19, 25]. The advantage of Sprites is that they can robustly handle camera motion. In addition, image content that temporally leaves the field of view can be more efficiently represented. Sprites can be used to improve the efficiency of MCP in case of camera motion by warping a second prediction signal towards the actual frame. The technique first identifies background and foreground regions based on local motion estimates. Camera motion is then estimated on the background by applying parametric global motion estimation. After compensating for camera motion, the background content is integrated into a so-called *background mosaic*. The Sprite coder warps an appropriate segment of the background mosaic towards the current frame to provide the second reference signal. The motion model used is typically a six parameter affine model or an eight parameter perspective model. The generation of the background mosaic is conducted either on-line or off-line and the two approaches are referred to as *Dynamic Sprites* and *Static Sprites*, respectively. So far, only Static Sprites are part of the MPEG-4 standard [26]. For Static Sprites, an iterative procedure is applied to analyze the motion in a video sequences of several seconds to arrive at robust segmentation results. This introduces a delay problem that cannot be resolved in interactive applications. On the other hand, the on-line estimation problem for Dynamic Sprites is very difficult and only recently

some advantages have been reported [21].

An interesting generalization of the background memory and Sprite techniques has been proposed by WANG and ADELSON, wherein the image sequence is represented by *layers* [27]. In addition to the background, the so-called *layered coding* technique can represent other objects in the scene as well. As for Static Sprites, the layers are determined by an iterative analysis of the motion in a complete image sequence of several seconds.

A simplification of the clustering problem in object-based or region-based coding and the parameter estimation in Sprite and layered coding is achieved by restricting the motion compensation to one global model that compensates the camera motion and focal length changes [28, 29, 30]. Often, the background in the scene is assumed to be static and motion of the background in the image plane is considered as camera motion. For the *global motion compensation* of the background often an affine motion model is used where the parameters are estimated typically using two steps. In the first step, the motion parameters are estimated for the entire image and in the second step, the largest motion cluster is extracted. The globally motion-compensated frame is either provided additionally as a second reference frame or the prior decoded frame is replaced. Given the globally motion-compensated image as a reference frame, typically a block-based hybrid video coder conducts translational motion compensation. The drawback of global motion compensation is the limitation in rate-distortion performance due to the restriction to one motion parameter vector per frame. The benefits of this approach are the avoidance of sophisticated segmentation and parameter estimation problems. Global motion compensation is therefore standardized as an Annex of H.263+ [31] to enhance the coding efficiency for the on-line encoding of video.

In this paper, the global motion compensation idea is extended to employing several affine motion parameter vectors. The estimation of the various affine motion parameter vectors is conducted so as to handle multiple independently moving objects in combination with camera motion and focal length change. Long-term statistical dependencies are exploited as well by incorporating long-term memory MCP. The paper is organized as follows. In Section 2, the extension of long-term memory MCP to affine motion compensation is explained. The coder control is described in Section 3, where the estimation procedure for the affine motion parameters and the reference picture warping are presented. Then, the determination of the efficient number of affine motion parameter vectors is described. Finally, in Section 4, experimental results are presented that illustrate the improved rate-distortion performance in comparison to TMN-10 and long-term memory MCP.

2 Affine Multi-Frame Motion Compensation

In this section, the structure of the affine multi-frame motion compensation is explained. First, the extension of the multi-frame buffer by warped versions of decoded frames is described. Then, the necessary syntax extensions are outlined and the affine motion model, i.e., the equations that relate the affine motion parameters to the pixel-wise motion vector field are presented.

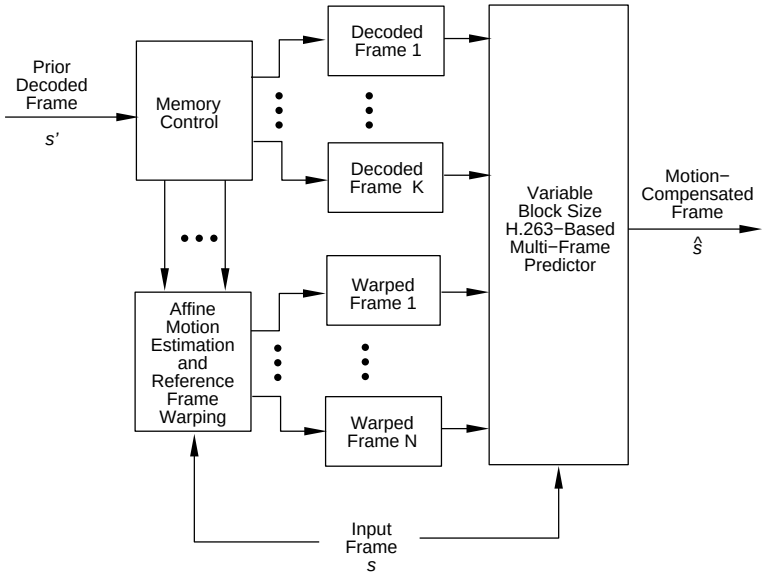

Figure 1: Blockdiagram of the affine multi-frame motion-compensated predictor.

The blockdiagram of the multi-frame affine motion-compensated predictor is depicted in Fig. 1. The motion-compensated predictor utilizes $M = K + N$ ($M \geq 1$) picture memories. The M picture memories are composed of two sets:

1. K past decoded frames and
2. 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 [31, 1]. The N warped reference frames are determined using the following two steps:

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

The number of efficient reference frames $M^* \leq M$ is determined by evaluating their rate-distortion efficiency for each reference frame. The M^* chosen reference frames with the associated affine motion parameter vectors are transmitted in the header of each picture. The order of their transmission provides an 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 affine motion compensation, the translational motion vector and the affine motion parameter vector are combined to obtain the displacement field for that image segment.

Figures 2 and 3 show an example for affine multi-frame warping. The left-hand side in Fig. 2 is the most recent decoded frame that would be the only frame to predict the right-hand side in Fig. 2 in single-frame motion compensation. Four out of the set of additionally employed reference frames are shown in Fig. 3. 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 picture reference parameter.

2.1 Syntax of the Video Codec

Affine multi-frame MCP is integrated into a video codec that is based on ITU-T Recommendation H.263+ [31]. H.263 uses the typical basic structure that has been predominant in all video coding standards since the development of H.261 [32] in 1990, where the image is partitioned into macroblocks of 16×16 luminance pixels and 8×8 chrominance pixels. Each macroblock can either be coded in INTRA or one of several predictive coding modes. In INTRA mode, the macroblock is further divided into blocks of size 8×8 pixels and each of these blocks is coded using DCT, scalar quantization, and run-level variable-length entropy coding. The predictive coding modes can either be of the types SKIP, INTER, or INTER+4V. For the SKIP mode, just one bit is spent to signal that the pixels of the macroblock are repeated from the prior coded frame. The INTER coding mode uses blocks of size 16×16 and the INTER+4V coding mode 8×8 pixels for motion compensation. For both modes, the MCP error image is encoded similarly to INTRA coding by using the DCT for 8×8 blocks, scalar quantization, and run-level variable-length entropy coding. The motion compensation can be conducted using half-pixel accurate motion vectors where the intermediate positions are obtained via bi-linear interpolation.

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. Hence, the proposed video codec enables the utilization of variable block-size coding, long-term memory prediction and affine motion compensation using such an adaptive method, where the use of the multiple reference frames and affine motion parameter vectors can be signaled with very

(a) (b)

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

Figure 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.

little overhead. The parameters for the chosen reference frames are transmitted in the header of each picture. First, their actual number M^* is signaled using a variable length code. Then, for each of the M^* reference frames, an index identifying one of the past K decoded pictures is transmitted. This index is followed by a bit signaling whether the indicated decoded frame is warped or not. If that bit indicates a warped frame, the corresponding six affine motion parameters are transmitted. This syntax allows the adaptation of the multi-frame affine coder to the source signal on a frame-by-frame basis without incurring much overhead. Hence, if affine motion compensation is not efficient, one bit is enough to turn it off.

2.2 Affine Motion Model

In this work an affine motion model is employed 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 vectors, the orthogonalization approach in [18] is adopted. 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 [18] a comparison was made to other approaches indicating the efficiency of the orthogonalized motion model. The motion model used for the investigations in this chapter is given as

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

in which x and y are discrete pixel locations in the image with $0 \le x < w$ and $0 \le y < h$ 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

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

The affine motion parameters a_i are quantized as follows

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

where $Q(\cdot)$ means rounding to the nearest integer value. The quantization levels of the affine motion parameters $q_i = \Delta \cdot \tilde{a}_i$ are entropy-coded and transmitted. It has been 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. Typically, an affine motion parameter vector requires between 8 and 40 bits for transmission.

3 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 of the video signal. 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, one additionally faces the problem that the number of reference frames has to be determined since each warped reference frame is associated to an overhead bit-rate. Therefore, the affine motion parameter vectors must be assigned to large image segments to keep their number small. In most cases however, these large image segments usually cannot be chosen so as to partition the image uniformly. The proposed solution to this problem is as follows:

A. Estimate N affine motion parameter vectors between the current and the K previous frames each corresponding to one of N initial clusters.

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 vectors.

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

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

In the following, steps A-D are described in detail.

3.1 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 successive coding of the resulting image segments. In

this work, an explicit segmentation of the scene is avoided. Instead, the image is partitioned into blocks of fixed size which are referred to as clusters in the following. For each cluster one affine motion parameter vector is estimated that describes the motion inside this cluster between a decoded frame and the current original frame. The estimation of the affine motion parameter vector for each cluster is conducted 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 parameter vectors.

For the first step, block matching in the long-term memory buffer is performed in order to robustly deal with large displacements yielding L translational 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 MSE. In the third step, the resulting affine motion parameter vector 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 MSE that is measured over the pixels of the cluster is chosen among the L considered candidates. In the following, the various steps are discussed in detail.

For the *first step*, the initial motion vector estimation, two approaches are discussed:

- *cluster-based initialization* and
- *macroblock-based initialization.*

For the *cluster-based initialization*, the MSE for block matching is computed over all pixels inside the cluster. The motion search proceeds over the search range of ± 16 pixels and produces one motion vector per reference frame and cluster. Hence, the number of considered candidates per cluster L is equal to the number of decoded reference frames K. This approach provides flexibility in the choice of the cluster size and with that the number of clusters N. Hence, it will be used in Section 4 to analyze the trade-off between rate-distortion performance and complexity that is proportional to the number of initial clusters N since this number is proportional to the number of warped reference frames.

However, the *cluster-based initialization* approach produces a computational burden that increases as the number of decoded reference frames K grows since the affine refinement routine

has to be repeated for each initial translational motion vector. On the other hand, translational motion estimation has to be conducted anyway for 16×16 blocks in H.263 and the long-term memory MCP coder. Hence, the re-use of those motion vectors would not only avoid an extra block matching step for the initializations, it would also fix the number of initial motion vectors to the number of macroblocks per cluster. This approach is called the *macroblock-based initialization*. Therefore, an image partitioning is considered where the clusters are aligned with the macroblock boundaries. An example for such an initial partitioning 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. 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. If the motion vector of

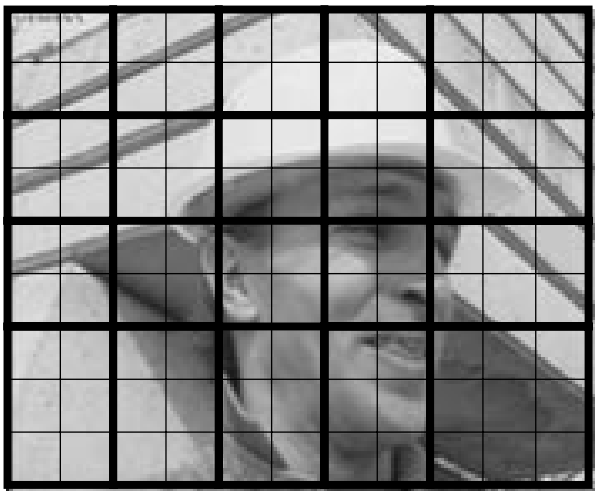

Figure 4: Image partitioning of a QCIF frame of the sequence *Foreman* into N =20 cluster.

each macroblock is utilized as an initialization to the affine refinement step, either $L = 4, 6$ or 9 candidates have to be considered. This number is independent from the number of decoded reference frames K.

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

$$\boldsymbol{m}^I = \underset{\boldsymbol{m} \in \mathcal{M}}{\operatorname{argmin}} \left\{ D_{DFD}(\boldsymbol{S}_k, \boldsymbol{m}) + \lambda_{MOTION} \cdot R(\boldsymbol{S}_k, mv). \right\} \tag{4}$$

The distortion $D_{DFD}(\boldsymbol{S}_k, \boldsymbol{m})$ for the 16×16 block $\boldsymbol{S}_k$ between the current frame $s[x, y, t]$ and the decoded reference frame $s'[x, y, t - m_t]$ is computed as

$$D_{DFD}(\boldsymbol{S}_k, \boldsymbol{m}) = \sum_{x,y \in \boldsymbol{\mathcal{B}}_k} \left(s[x, y, t] - s'[x - m_x, y - m_y, t - m_t] \right)^2 , \tag{5}$$

where $\boldsymbol{\mathcal{B}}_k$ is the set of pixel positions corresponding to the block $\boldsymbol{S}_k$. $R(\boldsymbol{S}_k, \boldsymbol{m})$ is the bit-rate associated with the motion vector. The minimization proceeds over the search space $\mathcal{M} = [-16 \ldots 16] \times [-16 \ldots 16] \times [0 \ldots K-1]$. First, the integer-pixel motion vectors are determined that minimize the Lagrangian cost term in (4) for each of the K reference frames. Then, these K integer-pixel accurate motion vectors are used as initialization of a half-pixel refinement step which tests the 8 surrounding half-pixel 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 [31].

For the *second step*, the affine refinement, 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 $s'[x, y, t - m_t]$ towards the current frame $s[x, y, t]$ as follows

$$\hat{s}[x, y, t] = s'[x - m_x^I, y - m_y^I, t - m_t^I]. \tag{6}$$

This motion compensation has to be conducted only for the pixels inside the considered cluster $\boldsymbol{\mathcal{A}}$. The minimization criterion for the affine refinement step reads as follows

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

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).

The signal $\hat{s}[x - m_x[\boldsymbol{a}, x, y], y - m_y[\boldsymbol{a}, x, y], t]$ is linearized around the spatial location (x, y) for small spatial displacements $(m_x[\boldsymbol{a}, x, y], m_y[\boldsymbol{a}, x, y])$ yielding

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

Hence, the error signal in (8) reads

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

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

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

with the abbreviations

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

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-compensated frame difference. In this work, the pseudo inverse technique is used which is implemented via singular value decomposition. The linearization (9) holds for small displacements only which might require an iterative approach to solve (11). However, due to the translational initialization and the subsequent quantization of the affine motion parameters it turns out that no iteration is needed. Experiments verify this statement, where the number of iterations have been varied without observing a significant difference in resulting rate-distortion performance.

The spatial intensity gradients are computed following [33, 34]. With $z \in \{x, y\}$ the spatial gradients are given as

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

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

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

The estimates provide the gradient of the point in-between the four samples and between the pre-compensated and the current image [34]. Since the spatial gradients are computed between the pixel positions, the frame difference $s[x,y,t] - \hat{s}[x,y,t]$ is computed as well 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 parameters for motion compensation between the reference frame $s'[x,y,t-m_t^I]$ and the current frame $s[x,y,t]$ is obtained via concatenating the initial translational motion vector $\boldsymbol{m}^I$ and the estimated affine motion parameter vector $\boldsymbol{a}^R$ yielding

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

The initial translational block matching and the affine refinement procedure are repeated for each of the L candidates. Finally, in the *fourth step*, the affine motion parameter vector is chosen that minimizes the MSE measured over the pixels in the cluster $\boldsymbol{\mathcal{A}}$.

3.2 Reference Picture Warping

For each of the N estimated affine motion parameter vectors, the corresponding reference frame is warped towards the current frame. The reference picture warping is conducted using the motion field that is computed via (1) given each affine motion parameter vector for the complete frame. Intensity values that correspond to non-integer displacements are computed using cubic spline interpolation [35] which turns out to be more efficient than bi-linear interpolation as the motion model becomes more sophisticated [36]. Hence, the multi-frame buffer is extended by N new reference frames that can be used for block-based prediction of the current frame as illustrated in Fig. 1.

3.3 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 a total of M reference frames. Our goal is to obtain a coded representation of the video frame that is efficient in terms of rate-distortion performance via choosing a combination of motion vectors, macroblock modes, and reference frames. Since affine multi-frame MCP is integrated into a hybrid video codec that is based on ITU-T Recommendation H.263+, we adapt the recommended encoding strategy TMN-10 [37] of H.263+ towards the new motion compensation approach.

The TMN-10 encoding strategy as proposed in [38] utilizes macroblock mode decision similar to [4]. For each macroblock, the coding mode with associated parameters is optimized given the decisions made for prior coded blocks only. Let the Lagrange parameter λ_{MODE} and the DCT quantizer value Q be given. The Lagrangian mode decision for a macroblock $\boldsymbol{S}_k$ in TMN-10 proceeds by minimizing

$$J_{\text{MODE}}(\boldsymbol{S}_k, I_k | Q, \lambda_{\text{MODE}}) = D_{\text{REC}}(\boldsymbol{S}_k, I_k | Q) + \lambda_{\text{MODE}} \cdot R_{\text{REC}}(\boldsymbol{S}_k, I_k | Q), \tag{17}$$

where the macroblock mode I_k is varied over the set $\{\text{INTRA}, \text{SKIP}, \text{INTER}\}$. Rate $R_{\text{REC}}(\boldsymbol{S}_k, I_k | Q)$ and distortion $D_{\text{REC}}(\boldsymbol{S}_k, I_k | Q)$ for the various modes are computed as follows.

For the INTRA mode, the 8×8 blocks of the macroblock $\boldsymbol{S}_k$ are processed by a DCT and subsequent quantization. The distortion $D_{\text{REC}}(\boldsymbol{S}_k, \text{INTRA} | Q)$ is measured as the SSD between the reconstructed and the original macroblock pixels. The rate $R_{\text{REC}}(\boldsymbol{S}_k, \text{INTRA} | Q)$ is the rate that results after run-level variable-length coding.

For the SKIP mode, distortion $D_{\mathrm{REC}}(\boldsymbol{S}_k, \mathrm{SKIP})$ and rate $R_{\mathrm{REC}}(\boldsymbol{S}_k, \mathrm{SKIP})$ do not depend on the DCT quantizer value Q of the current picture. The distortion is determined by the SSD between the current picture and each of the $M = K + N$ reference pictures for the macroblock pixels, and the rate is given as one bit per macroblock plus the number of bits necessary to signal the corresponding reference picture. Finally, that reference picture is chosen, for which the SKIP mode provides the smallest cost when evaluating (17).

The computation of the Lagrangian costs for the INTER coding mode is much more demanding than for INTRA and SKIP. This is because of the block motion estimation and motion compensation step. In order to produce the MCP signal, multi-frame block-based motion compensation is conducted. That is, half-pixel 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 employed, the corresponding motion vectors can be re-used. Otherwise, motion estimation over the K decoded frames has to be conducted as described for the *macroblock-based initialization*. When searching a warped reference frame, only a range of $[-2 \ldots 2] \times [-2 \ldots 2]$ spatially displaced pixels is considered. This small search range is justified by the fact that the warped frames are already motion-compensated and experiments with a larger search range show that only a very small percentage of motion vectors is found outside the $[-2 \ldots 2] \times [-2 \ldots 2]$ range. The resulting prediction error signal is similar to the INTRA mode processed by a DCT and subsequent quantization. The distortion D_{REC} is also measured as the SSD between the reconstructed and the original macroblock pixels. The rate R_{REC} is given as the sum of the bits for the motion vector and the bits for the quantized and run-level variable-length encoded DCT coefficients.

Finally, the best coding mode is chosen for each macroblock. During the minimization, the values that correspond to the best coding mode for a given reference frames are stored in an array. This is done to permit a fast access to the Lagrangian costs for the following step, where the number of efficient reference frames is determined.

3.4 Determination of the Number of Efficient 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 Lagrangian costs. The greedy optimization algorithm proceeds as follows:

1. Sort the $M = K + N$ reference frames according to the frequency of their selection.

2. Starting with the least popular frame, test the efficiency 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 the variable length code 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 corresponding index that is used to specify a particular reference frame using the block-based motion vectors. This index is entropy-coded using a variable length code and the sorting matches the selection statistics to the length of the code words.

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. Only the more popular frames are considered because they potentially correspond to a smaller rate and because of the goal to obtain a reduced number of reference frames in the end. 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.

4 Experiments

Within the framework of the multi-frame affine motion coder there are various free parameters that can be adjusted. In this section, empirical justifications are given for important parameter choices made. Attention is given to parameters that have the largest impact on the trade-off between rate-distortion performance and computational complexity. Regarding the affine motion coder, the important question about the number of initial clusters N is discussed. 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 investigated and the gains when combining affine and long-term memory MCP are presented.

4.1 Affine Motion Compensation

In this section, the parameter setting for the affine motion coder is investigated. For that, the warping is restricted 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 clusters N. The translational motion vector estimation is conducted using the *cluster-based initialization* as described in Section 3.1. The coder is initialized with $N = 1, 2, 4, 8, 16, 32, 64$, and 99 clusters. The partition into the N initial clusters is conducted 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 3.1. The estimated affine motion parameter vectors are used to warp the previous decoded frame N times as explained in Section 3.2. Block-based multi-frame motion estimation and determination of the number of efficient affine motion parameter vectors is conducted as described in Sections 3.3 and 3.4.

The left-hand side of Fig. 5 shows the average bit-rate savings for the set of test sequences summarized in Tab. 1 in the appendix. For comparison, rate-distortion curves have been generated and the bit-rate is measured at equal PSNR. The intermediate points of the rate-distortion curves are interpolated and the bit-rate that corresponds to a given PSNR value is obtained. The percentage in bit-rate savings corresponds to different absolute bit-rate values for the various sequences. Hence, also rate-distortion curves are shown. 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 against TMN-10 are very similar for the three different levels of reproduction quality. The number of initial clusters has a significant impact on resulting rate-distortion performance. The increase in bit-rate savings saturates for a large number of clusters, i.e., more than 32 clusters, reaching the value of 17 % for the set of test sequences considering the reproduction quality of 34 dB PSNR.

This can be explained when investigating the average number of affine motion parameter vectors are determined to be efficient and hence transmitted shown on right-hand side of Fig. 5. The curves for the average number of transmitted affine motion parameter vectors is generated with a similar method as the average bit-rate savings for a given PSNR value. The average number of affine motion parameter vectors increases with increasing average PSNR as well as an increased number of initial clusters. This is because the size of the measurement window becomes smaller as the number of initial clusters increases and the affine motion parameters are more accurate inside the measurement window. Hence, the coder chooses to transmit more affine motion parameter vectors. For very small numbers of initial clusters, a large percentage of the

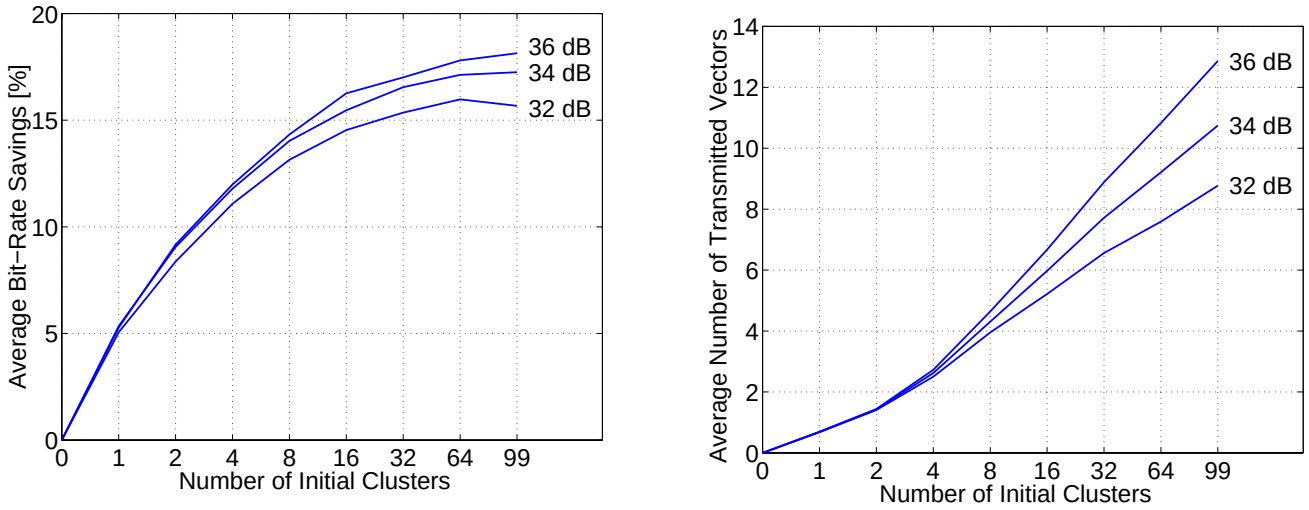

Figure 5: Average bit-rate savings against TMN-10 (left) and average number of transmitted affine motion parameter vectors (right) vs. number of initial clusters for the test sequences in Tab. 1 and three different levels of reproduction quality.

maximum number of affine motion parameter vectors is chosen. However, as the number of initial clusters is increased, a decreasing percentage of affine motion parameter vectors is transmitted.

Figure 6 shows the average bit-rate savings against TMN-10 at a reproduction quality of 34 dB PSNR for the set of test sequences where the result for each sequence is shown. The abbreviations *fm, mc, st, te, cs, md, nw,* and *si* correspond to those in Tab. 1. The solid line depicts the average bit-rate savings for the 8 test sequences at equal PSNR of 34 dB. The results differ quite significantly among the sequences in the test set. On the one hand, for the sequence *Silent Voice*, only a bit-rate saving of 6 % 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.

In Fig. 6, the asterisk shows the average result for the *macroblock-based initialization* of the affine estimation (see Section 3.1). Please recall that all experiments that were described so far are conducted using the *cluster-based initialization* for the translational motion vector estimation to have a simple means for varying the number of initial clusters. For the *macroblock-based initialization*, the segmentation in Fig. 4 is employed resulting in $N = 20$ clusters. The bit-rate saving of 15 % 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

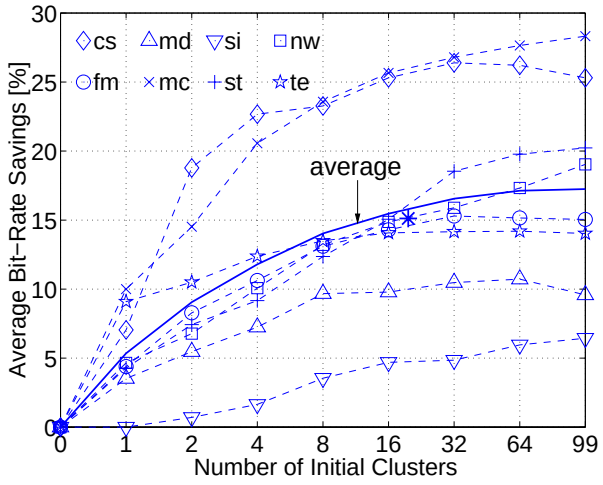

Figure 6: Average bit-rate savings against TMN-10 at 34 dB PSNR versus number of initial clusters for the test sequences in Tab. 1. For these results, only the prior coded picture is warped.

second for affine motion estimation, and the warping also takes 1 second. The pre-computation of the costs for the INTER, SKIP, and INTRA mode takes 2 seconds, and the remaining steps use 2 seconds. As a comparison, the TMN-10 coder which has a similar degree of run-time optimization uses 2 seconds per QCIF frame.

Finally, rate-distortion curves are depicted to evaluate the performance of this approach. For that, the DCT quantization parameter has been varied over values $Q = 4, 5, 7, 10, 15$, and 25 when encoding the sequences *Foreman*, *Mobile & Calendar*, *News*, and *Tempete*. The results are shown in Fig. 7, where the rate-distortion curves for the affine motion coder are compared to those of TMN-10 when running both codecs according to the conditions in Tab. 1. The following abbreviations indicate the two codecs compared:

- **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 $N = 20$ initial clusters using the *macroblock-based initialization*.

The PSNR gains vary for the different test sequences and tend to be larger as the bit-rate increases. In contrast, the relative bit-rate savings are more or less constant over the entire range of bit-rates that was tested. Typically, a PSNR gain of 1 dB compared to TMN-10 is obtained. The PSNR gains are up to 2.3 dB for the sequence *Mobile & Calendar*.

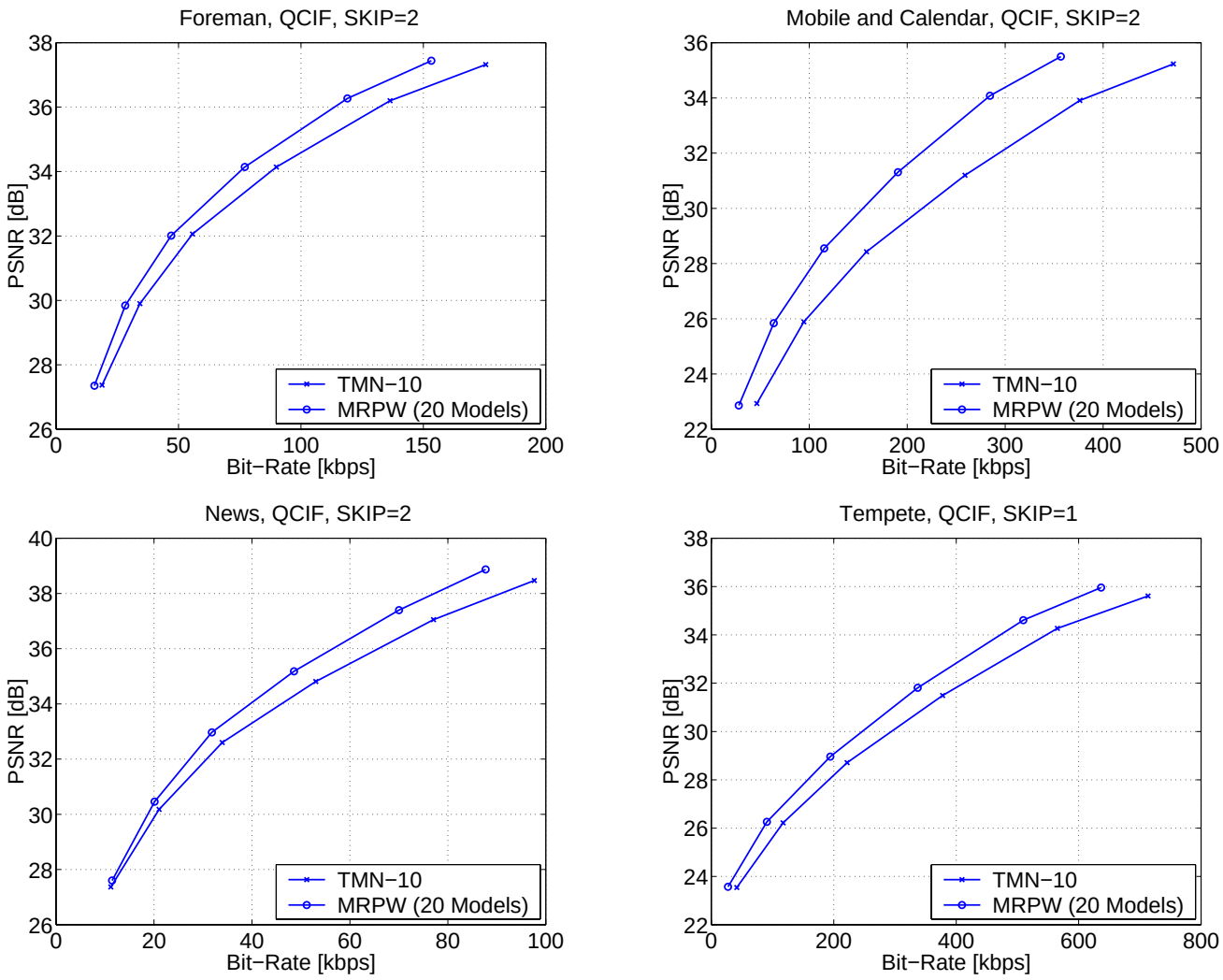

Figure 7: PSNR vs. overall bit-rate for the QCIF sequences *Foreman* (top left), *Mobile & Calendar* (top right), *News* (bottom left), and *Tempete* (bottom right).

4.2 Combination of Affine and Long-Term Memory Motion Compensation

In the previous section, it is shown that affine motion compensation provides significant bit-rate savings against TMN-10. The gains for the affine motion coder increase with an increasing number of initial clusters. A saturation of the gains is reported when increasing the number of initial clusters beyond 32. The number of initial clusters determines the number of reference frames that are warped. Hence, a parameter choice is proposed where 20 initial clusters are utilized providing an average bit-rate saving of 15 %.

In contrast to the affine motion coder where warped versions of the prior decoded frame are employed, the long-term memory MCP 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. In [6], the average bit-rate savings against TMN-10 at 34 dB PSNR for the set of test sequences that are achieved with the long-term memory MCP codec are presented. 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 against TMN-10. In [6], it is found that long-term memory MCP with 10 past decoded frames for most sequences yields a good compromise between complexity and bit-rate savings. Hence, we will use 10 decoded frames when combining long-term memory prediction and affine motion compensation.

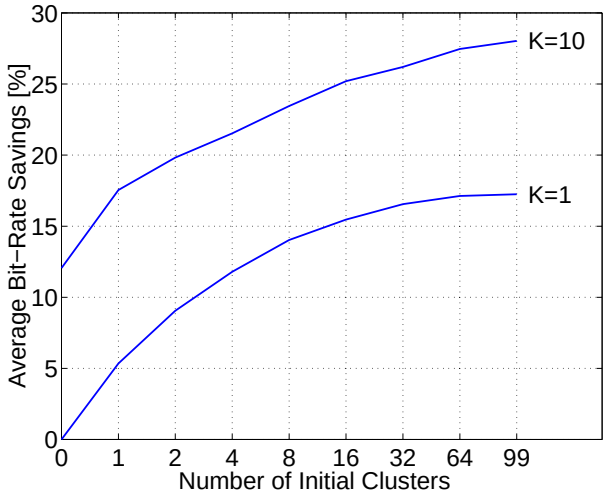

Figure 8: Average bit-rate savings against TMN-10 at 34 dB PSNR versus number of initial clusters for the set of test sequences in Tab. 1. Two cases are shown: (*i*) affine warping using $K = 1$ reference frames (lower solid curve) and (*ii*) affine warping using $K = 10$ reference frames (upper solid curve).

In Fig. 8, the result is depicted when combining the affine motion coder and long-term memory MCP. This plot shows average bit-rate savings against TMN-10 at 34 dB PSNR versus the number of initial clusters for the set of test sequences in Tab. 1. Two cases are shown:

i) affine warping using $K = 1$ reference frame (lower solid curve),

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 on the left-hand side in Fig. 5. To obtain the result for the case $K = 10$, the combined coder is run using the *cluster-based initialization* with $N = 1, 2, 4, 8, 16, 32, 64,$ and 99 initial 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 decoded frames (see Section 3.1). 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.

Figure 9 shows the bit-rate savings against TMN-10 for each of the test sequences in Tab. 1 when employing $K = 10$ decoded reference frames versus the number of initial clusters N using dashed lines. The bit-rate savings are more than 35 % for the sequences *Container Ship* and *Mobile & Calendar* when using 32 or more initial clusters. Interestingly, when using $K = 10$ reference frames and 16 or more initial clusters the bit-rate savings are never below 17 %.

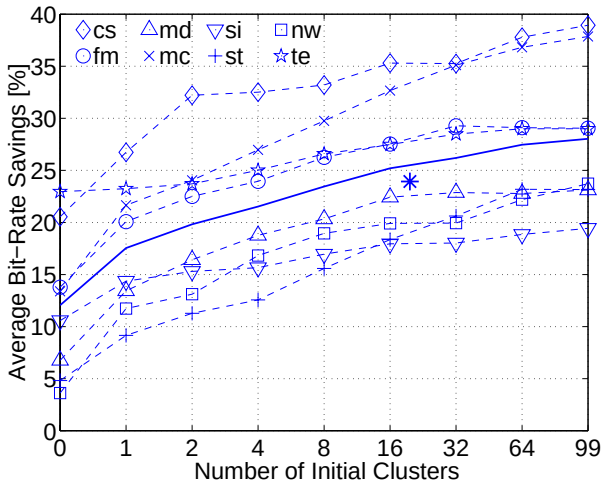

Figure 9: Average bit-rate savings against TMN-10 at 34 dB PSNR versus number of initial clusters for the set of test sequences in Tab. 1. For these results, the $K = 10$ past decoded frames may be utilized for warping.

In Fig. 9, the asterisk shows the result for the case of *macroblock-based initialization*. For that, the initial segmentation in Fig. 4 is employed. The initial motion vectors for the affine motion estimation are those best matches found for the macroblocks in each cluster when searching $K = 10$ decoded reference frames. An average bit-rate saving of 24 % is obtained for the set of 8 test sequences in Tab. 1.

The measured bit-rate savings correspond to PSNR gains up to 3 dB. Figure 10 shows rate-

distortion curves for the four test sequences *Foreman*, *Mobile & Calendar*, *Container Ship*, and *Silent Voice*. The curves depict the results that are obtained with the following three codecs:

- **TMN-10:** The H.263 test model using Annexes D, F, I, J, and T.
- **LTMP:** As TMN-10, but motion compensation is extended to long-term memory prediction with $K = 10$ decoded reference frames.
- **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 initial clusters is $N = 20$ and the *macroblock-based initialization* is employed.

Long-term memory MCP with $K = 10$ frames and without affine warping is always better than TMN-10 as already demonstrated in [5, 6]. Moreover, long-term memory MCP in combination with affine warping is always better than the case without affine warping. Typically, bit-rate savings between 20 and 35 % can be obtained which correspond to PSNR gains of 2-3 dB. For some sequences long-term memory prediction provides the most gain (*Silent Voice*) while for other sequences the affine motion coder is more important (*Mobile & Calendar*).

For the sequence *Mobile & Calendar* the gap between the result for the long-term memory MCP codec with and without affine motion compensation is visible for the lowest bit-rates as well. This results in a bit-rate saving of 50 %. Moreover, for some sequences, the gain obtained by the combined coder is larger than the added gains of the two separate coders. For example, the long-term memory prediction gain for *Mother & Daughter* is 7 % for $K = 10$ reference pictures when measuring over all coded frames. The gain obtained for the affine motion coder is 10 % when using 32 initial clusters. However, the combined coder achieves 23 % bit-rate savings for the sequence *Mother & Daughter*.

5 Conclusions

The idea of reference picture warping can be regarded as an alternative approach to assigning affine motion parameters to large image segments with the aim of a rate-distortion efficient motion representation. Although the affine motion parameter vectors are determined on sub-areas of the image, they can be employed at any position inside the frame. Instead of performing a joint estimation of the image partition and the associated affine motion parameter vectors, reference frames are warped and selected in a rate-distortion efficient way on a block basis. Hence, the presented approach decomposes the joint optimization task of finding an efficient combination of affine motion parameters, regions and other parameters into separate steps. Each of these steps

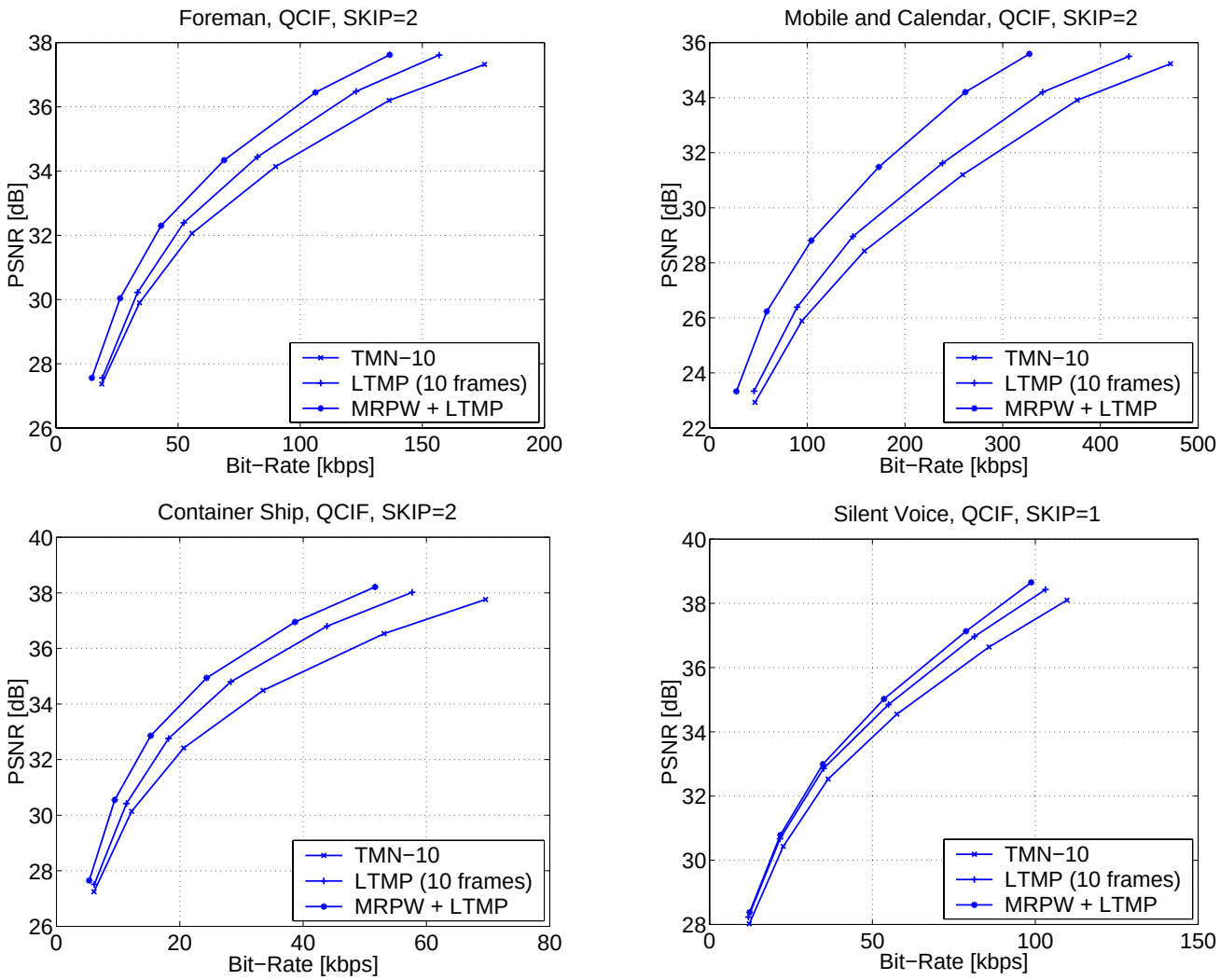

Figure 10: PSNR vs. overall bit-rate for the QCIF sequences *Foreman* (top left), *Mobile & Calendar* (top right), *Container Ship* (bottom left), and *Silent Voice* (bottom right).

takes an almost constant amount of computation time which is independent of the context of the input data. The coder robustly adapts the number of affine motion parameter vectors 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 combined affine and long-term memory MCP codec is an example for an efficient multi-frame video compression scheme. The two incorporated multi-frame concepts seem to complement each other well providing almost additive rate-distortion gains. When warping the prior decoded frame, average bit-rate savings of 15 % against TMN-10 are reported for the case that 20 warped reference pictures are used. For the measurements, reconstruction PSNR is identical

to 34 dB for all cases considered. These average bit-rate savings are measured over a set of 8 test sequences that represent a large variety of video content. Within the test set, the bit-rate savings vary from 6 to 25 %. Long-term memory prediction has been already demonstrated as an efficient means to compress motion video [5, 6]. 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 warped reference pictures and 10 decoded reference frames, average bit-rate savings of 24 % can be obtained for the set of 8 test sequences. The minimal bit-rate savings inside the test set are 15 % while the maximal bit-rate savings are reported to be up to 35 %. These bit-rate savings correspond to gains in PSNR between 0.8 and 3 dB. For some cases, the combination of affine and long-term memory MCP provides more than additive gains.

Appendix A: Test Sequences

The experiments in this paper are conducted using the QCIF test sequences and conditions in Tab. 1. The sequences and test conditions are almost identical to those that are maintained by the ITU-T Advanced Video Coding Experts Group. This set of sequences has been chosen so as to represent a wide variety of statistical dependencies and different types of motion and texture.

Sequence Name	Abbreviation	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 Voice	si	300	1	No

Table 1: Test sequences and simulation conditions.

The first four sequences contain a large amount of motion including a moving camera position and focal length change. The last four sequences are low motion sequences with a fixed camera. This set was chosen so as to cover a broad range of possible scenes that might occur in applications such as video conferencing or video streaming.

In all experiments, bit-streams are generated that are decodable producing the same PSNR values at encoder and decoder. The first frame of the image sequence is coded in INTRA mode followed by INTER-coded pictures. In INTER pictures the macroblocks can either be coded predictively using one of the INTER macroblock modes or as INTRA blocks. In the simulations, the first intra-coded frame is identical for all cases considered.

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] 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*, Phoenix, AZ, USA, Dec. 1991, pp. 85–90.

[3] B. Girod, "Rate-Constrained Motion Estimation", in *Proceedings of the SPIE Conference on Visual Communications and Image Processing*, Chicago, IL, USA, Sept. 1994, vol. 2308, pp. 1026–1034.

[4] 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.

[5] 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.

[6] T. Wiegand and B. Girod, *Multi-Frame Motion-Compensated Prediction for Video Transmission*, Kluwer Academic Publishers, 2000, in preparation.

[7] 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.

[8] 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.

[9] M. Hötter and R. Thoma, "Image Segmentation Based on Object Oriented Mapping Parameter Estimation", *Signal Processing: Image Communication*, vol. 15, no. 3, pp. 315–334, Oct. 1988.

[10] N. Diehl, "Object-Oriented Motion Estimation and Segmentation in Image Sequences", *Signal Processing: Image Communication*, vol. 3, no. 1, pp. 23–56, Jan. 1991.

[11] 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.

[12] 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.

[13] 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, vol. 2727, pp. 1152–1163.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] M. Karczewicz, J. Niewęgłowski, and P. Haavisto, "Video Coding Using Motion Compensation with Polynomial Motion Vector Fields", *Signal Processing: Image Communication*, vol. 10, no. 3, pp. 63–91, July 1997.

[19] F. Dufaux and F. Moscheni, "Background Mosaicking for Low Bit Rate Video Coding", in *Proceedings of the IEEE International Conference on Image Processing*, Lausanne, Switzerland, Sept. 1996, vol. 3, pp. 673–676.

[20] ISO/IEC JTC1/SC29/WG11 MPEG96/N1648, "Core Experiment on Sprites and GMC", Apr. 1997.

[21] A. Smolic, T. Sikora, and J.-R. Ohm, "Long-Term Global Motion Estimation and Its Application for Sprite Coding, Content Description, and Segmentation", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no. 8, pp. 1227–1242, Dec. 1999.

[22] N. Mukawa and H. Kuroda, "Uncovered Background Prediction in Interframe Coding", *IEEE Transactions on Communications*, vol. 33, no. 11, pp. 1227–1231, Nov. 1985.

[23] D. Hepper, "Efficiency Analysis and Application of Uncovered Background Prediction in a Low Bit Rate Image Coder", *IEEE Transactions on Communications*, vol. 38, no. 9, pp. 1578–1584, Sept. 1990.

[24] X. Yuan, "Hierarchical Uncovered Background Prediction in a Low Bit-Rate Video Coder", in *Proceedings of the Picture Coding Symposium*, Lausanne, Switzerland, Mar. 1993, p. 12.1.

[25] K. Zhang and J. Kittler, "A Background Memory Update Scheme for H.263 Video Codec", in *Proceedings of the European Signal Processing Conference*, Island of Rhodes, Greece, Sept. 1998, vol. 4, pp. 2101–2104.

[26] ISO/IEC JTC1/SC29/WG11 MPEG98/N2202, "Committee Draft", Mar. 1998.

[27] J. Y. A. Wang and E. H. Adelson, "Representing Moving Images with Layers", *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 625–638, Sept. 1994.

[28] M. Hötter, "Differential Estimation of the Global Motion Parameters Zoom and Pan", *Signal Processing*, vol. 16, no. 3, pp. 249–265, Mar. 1989.

[29] 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.

[30] ISO/IEC JTC1/SC29/WG11 MPEG96/M1686, "Core experiment on global motion compensation", Submitted to Video Subgroup, Feb. 1997.

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

[32] ITU-T Recommendation H.261, "Video Codec for Audiovisual Services at $p \times 64$ kbit/s", Mar. 1993.

[33] B. K. P. Horn and B. G. Schunck, "Determining Optical Flow", *Artificial Intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.

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

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

[36] 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, no. 1, pp. 105–127, Mar. 1995.

[37] 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/q15d65.doc, Apr. 1998.

[38] 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.