ADAPTIVE MOTION MODEL SELECTION USING A CUBIC SPLINE BASED ESTIMATION FRAMEWORK

Haricharan Lakshman[1], *Heiko Schwarz*[1] *and Thomas Wiegand*[1,2]

[1]Image Processing Department
Fraunhofer Institute for Telecommunications - Heinrich Hertz Institute
Einsteinufer 37, 10587 Berlin, Germany

[2]Image Communication Chair
Department of Telecommunication Systems
Technical University of Berlin
Einsteinufer 17, 10587 Berlin, Germany

ABSTRACT

A block based video coder that supports multiple motion models is proposed. Apart from the typical translational motion model, we employ parametric models to more accurately represent complex motions that occur in video sequences. A novel method for estimating the warping parameters in a rate-constrained way is presented. A cubic spline framework is utilized to obtain fractional-accuracy samples for motion compensation. Efficient motion vector prediction schemes are developed to maintain the continuity of the predictor in spite of different motion models. Bit rate savings up to 11.7% in IPPP and 9.3% in Hierarchical B pictures are shown compared to an improved H.264/AVC reference.

Index Terms— Motion Compensated Prediction, Parametric motion models, Cubic Spline Interpolation

1. INTRODUCTION

Motion-compensation (MC) using translational motion models is well established in international standards like the H.264/AVC [1]. Increasing the number of motion parameters has the potential to improve compression efficiency. Some of the early proposals for the H.264/AVC standard were based on affine motion models [2, 3]. A technique for using an affine model to generate additional reference pictures for MC was proposed in [4]. The approach in [4] effectively combined translational and affine motion models within one picture, however, motion clustering into regions is generally difficult in case of complex motions.

The advantages of using PMMs can be viewed in two perspectives. From a motion estimation point of view, PMMs are able to more accurately represent complex motions that occur in typical video sequences, whereas from an optimization point of view, the additional parameters provide extra degrees of freedom to reduce the rate-distortion cost. Although PMMs show gains [5] when applied on previous standards like the H.263, the advantage might be reduced in H.264/AVC because of the improved structure. In order to fully utilize H.264/AVC capabilities and harness the advantages of PMMs, we propose to include the PMM as a candidate for MC prediction, which competes with the traditional translational model for each block. This involves the transmission of the chosen prediction model and the extra motion parameters whenever applicable.

The motion vector field when using PMM can consist of locations in the reference picture that do not fall on the integer sample grid. In order to get the prediction signal, it is necessary to evaluate the reference picture at non-grid locations. To this end, discrete-to-continuous mapping techniques [6] are utilized. In [3], the intermediate locations are evaluated using a cubic convolution kernel that is fitted to reference samples. In the H.264/AVC standard, a 6-tap filter is used for estimating the sample value at half-sample positions and a bi-linear filter is used for quarter-sample estimation. Compared to the H.264/AVC filters, the cubic convolution kernel (4-tap) shows a loss of RD performance for quarter-sample interpolation. Hence we propose to use cubic splines for the interpolation and discrete-to-continuous mapping. The spline coefficients are signal adaptive and are known to provide very good interpolation properties [6]. Therefore, even when PMMs are not used for MC, cubic splines are a good candidate for interpolation of quarter-sample locations in a translational model.

2. VIDEO CODING FRAMEWORK

We use a quad-tree framework in which the block sizes are allowed to vary from 64×64 to 4×4. There has been a considerable gain reported [7] when using block sizes higher than the traditional 16×16, especially for HD material. The encoder starts with the biggest block size, evaluates the RD cost of all possible prediction modes and chooses the mode with the smallest cost. The block is then split into four sub-blocks and the RD costs are evaluated for each sub-block. The encoder then compares the cost of the parent block with the sum of the costs of the sub-blocks to decide whether to split or not. Each sub-block can be further split recursively until the minimum allowed block size. At the end, a quad-tree prediction

structure is obtained with the chosen prediction mode for each block. The transmission of translational MV and Intra mode direction is same as in H.264/AVC. In case of the parametric mode, extra 'warp' parameters are transmitted, described in detail in Sec. 4.

Cubic spline based discrete-to-continuous mapping

Splines are piecewise polynomials with pieces that are smoothly connected together. Within the family of polynomial splines, cubic splines are the most popular in applications because of their minimum curvature property [6]. B-splines are the basic building blocks for splines. Denoting the B-spline of degree 3 as $\beta(x)$ and the spline coefficients as $w(k)$, the cubic spline model for a particular location x in the signal s is given by,

$$s(x) = \sum_{k}^{k+3} w(k)\ \beta(x-k), \tag{1}$$

where $k = \lceil x - 2 \rceil$. Given the signal samples, the interpolation task is to determine the spline coefficients such that there is a perfect fit at integer locations. This computation of the spline coefficients can be implemented very efficiently using a cascade of first order causal and anti-causal recursive filters [6]. The spline coefficient computation is done in a separable way i.e. 1D filtering is applied successively along the rows and columns of the reference image. These spline coefficients are further used for gradient computations at arbitrary fractional positions during motion estimation. Using the same spline model for these different aspects of interpolation and motion estimation makes the design robust.

3. PARAMETRIC MOTION MODEL ESTIMATION

In a translational motion model, the displacement of all samples in a block is the same. On the other hand, the displacements of the samples in a block can be made to vary according to some parameters in case of PMMs. The displacement of a location $\boldsymbol{x} = [x, y]^T$ resulting from a parametric motion model $\boldsymbol{M}(\boldsymbol{x})$ with parameter vector $\boldsymbol{c}$ can be represented as,

$$\boldsymbol{d}(\boldsymbol{x}) = \boldsymbol{M}(\boldsymbol{x}) \cdot \boldsymbol{c}. \tag{2}$$

For instance, the displacement due to an affine motion is,

$$\begin{bmatrix} d_x(x,y) \\ d_y(x,y) \end{bmatrix} = \begin{bmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{bmatrix} \cdot [c_0 \cdots c_5]^T \tag{3}$$

Let s denote the original frame and s' the reference frame, then the prediction error can be expressed as,

$$\begin{aligned} u(\boldsymbol{x}) &= s(\boldsymbol{x}) - s'(\boldsymbol{x} + \boldsymbol{d}(\boldsymbol{x})) \\ &= s(\boldsymbol{x}) - s'(\boldsymbol{x} + \boldsymbol{M}(\boldsymbol{x}) \cdot \boldsymbol{c}). \end{aligned} \tag{4}$$

The task of motion estimation is to find the optimal vector $\boldsymbol{c}$ such that the prediction error for the block is minimized. To this end, the Gauss-Newton method for minimization of non-linear equations is utilized. It is an iterative procedure in which each iteration a linearization of the problem around the current estimate is performed and the the simplified problem is then solved.

Although rate-constrained estimation of translational MV is well-known, the standard techniques so far typically estimate the parameters of PMMs by minimizing the prediction error without regarding the rate for transmitting the extra parameters. We propose a novel method which performs an approximately rate-constrained estimation of the parameter set. We compute a predictor $\boldsymbol{p}$ for the motion parameters (using parameters from neighboring blocks) and try to minimize the motion estimation error while keeping the estimated vector $\boldsymbol{c}$ close to the predicted value.

Approximately rate-constrained estimation

The algorithm starts by performing a block matching of the current block with the reference picture. The resulting motion vector is utilized to initialize the translational component of the parameter vector $\boldsymbol{c}$ and the higher order components of $\boldsymbol{c}$ are set to 0. In each iteration, a vector $\boldsymbol{\Delta c}_i$ is estimated and is used to update the solution,

$$\boldsymbol{c} = \boldsymbol{c}_i + \boldsymbol{\Delta c}_i \tag{5}$$

Assuming the update vector $\boldsymbol{\Delta c}_i$ is small, the linearization of the reference picture model can be performed as,

$$\begin{aligned} s'(\boldsymbol{x} + \boldsymbol{d}(\boldsymbol{x})) &= s'(\boldsymbol{x} + \boldsymbol{M}(\boldsymbol{x}) \cdot \boldsymbol{c}) \\ &= s'(\boldsymbol{x} + \boldsymbol{M}(\boldsymbol{x}) \cdot \boldsymbol{c}_i + \boldsymbol{M}(\boldsymbol{x}) \cdot \boldsymbol{\Delta c}_i) \\ &\approx s'(\boldsymbol{x} + \boldsymbol{M}(\boldsymbol{x}) \cdot \boldsymbol{c}_i) + \\ &\quad \boldsymbol{M}(\boldsymbol{x}) \cdot \boldsymbol{\Delta c}_i \cdot \frac{\partial s'}{\partial \boldsymbol{x}}\Big|_{\boldsymbol{x} + \boldsymbol{M}(\boldsymbol{x}) \cdot \boldsymbol{c}_i}. \end{aligned} \tag{6}$$

Denoting

$$\boldsymbol{x}_i = \boldsymbol{x} + \boldsymbol{M}(\boldsymbol{x}) \cdot \boldsymbol{c}_i \tag{7}$$

$$\boldsymbol{g}(\boldsymbol{x}_i) = \boldsymbol{M}(\boldsymbol{x}) \cdot \frac{\partial s'}{\partial \boldsymbol{x}}\Big|_{\boldsymbol{x} + \boldsymbol{M}(\boldsymbol{x}) \cdot \boldsymbol{c}_i} \tag{8}$$

the prediction error can now be written as,

$$\begin{aligned} u(\boldsymbol{x}) &\approx s(\boldsymbol{x}) - s'(\boldsymbol{x}_i) - \boldsymbol{g}(\boldsymbol{x}_i)\boldsymbol{\Delta c}_i \\ &= u(\boldsymbol{x}_i) - \boldsymbol{g}(\boldsymbol{x}_i)\boldsymbol{\Delta c}_i, \end{aligned} \tag{9}$$

where, the difference $s(\boldsymbol{x}) - s'(\boldsymbol{x}_i)$ is denoted by $u(\boldsymbol{x}_i)$.

Considering a set of samples $\boldsymbol{x} \in \mathcal{A}$ to represent a block in a video frame and representing the block of $u(\boldsymbol{x}_i)$ as a vector $\boldsymbol{u}$ and the block of $\boldsymbol{g}(\boldsymbol{x}_i)$ as a matrix $\boldsymbol{G}$, the error energy becomes,

$$J_1 = (\boldsymbol{u} - \boldsymbol{G} \cdot \boldsymbol{\Delta c}_i)^T (\boldsymbol{u} - \boldsymbol{G} \cdot \boldsymbol{\Delta c}_i). \tag{10}$$

The difference energy between the estimated motion parameters and the predicted parameters can be written as,

$$\begin{aligned} J_2 &= (\boldsymbol{c}_i + \boldsymbol{\Delta c}_i - \boldsymbol{p})^T (\boldsymbol{c}_i + \boldsymbol{\Delta c}_i - \boldsymbol{p}) \\ &= (\boldsymbol{\Delta c}_i - \boldsymbol{q})^T (\boldsymbol{\Delta c}_i - \boldsymbol{q}). \end{aligned} \tag{11}$$

Assigning a weight of λ to the parameter difference energy, we define the overall cost function for the optimization as,

$$J = J_1 + \lambda \cdot J_2. \tag{12}$$

This cost function can be minimized by setting its partial derivatives with respect to $\boldsymbol{\Delta c}_i$ to zero, giving

$$(\boldsymbol{G}^T\boldsymbol{G} + \lambda I) \cdot \boldsymbol{\Delta c}_i = (\boldsymbol{G}^T\boldsymbol{u} + \lambda \boldsymbol{q}). \tag{13}$$

Solving this linear system of equations results in the update vector $\boldsymbol{\Delta c}_i$. The new motion parameters are computed and the entire procedure is repeated till the maximum number of iterations is reached. It has to be noted that the described procedure only finds the local optimum because the actual cost function (without linearization) is not necessarily convex. In certain cases, the update $\boldsymbol{\Delta c}_i$ cannot be computed (when rank of matrix in Eq. 13 becomes low, especially for small blocks). Therefore, after each possible update based on the linearized model, the actual error energy is computed. Whenever the error increases compared to the previously computed parameters, the iteration is stopped and the last valid estimate is used for MC prediction.

Although the second term J_2 does not exactly represent the rate of the motion parameter, is provides a bias and a regularization of the estimate with a similar result as our test on translational motion vectors have shown.

4. TRANSMISSION OF MOTION PARAMETERS

In order to introduce the option of multiple motion models for each block, it is necessary to take care of the continuity of the MV predictors across different motion models. Before performing motion parameter prediction, we map them to a suitable space because directly using the estimated parameters is not efficient for transmission. We choose the affine motion model as an example of PMM and elaborate the next steps. A direct representation of $c_0, c_1, \cdots, c_5$ suffers from the fact that c_2, c_5 (refer Eq. 3) do not really represent the translational displacement of the entire block, but that of the top-left sample in that block.

4.1. Transmission of Translational Component

In [5], it is proposed to orthonormalize $\boldsymbol{c}$ before quantization. Although, orthonormalization would make the parameters more robust to quantization effects, it cannot be directly used in prediction because of different MV quantization steps. Hence, we propose to extract the translational shift of the centroid of the current block and quantize it using the same step size as regular MVs so that it can be predicted using the MVs of neighboring blocks. The warping parameters c_0, c_1, c_3, c_4 are used to compute the displacement of the top-left and top-right samples relative to the centroid of the considered block. Such a mapping to corner motion vectors (CMV) enables us

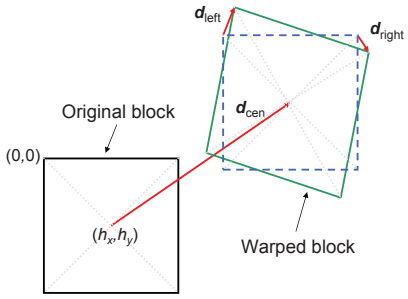

Fig. 1. *Example showing the original block (black) and the warped block (green). The translation of the centroid $\boldsymbol{d}_{cen}$ is shown by the long red MV. Using $\boldsymbol{d}_{cen}$, the decoder obtains the dotted blue block. The relative warping with respect to the centroid, $\boldsymbol{d}_{left}$ and $\boldsymbol{d}_{right}$, are depicted as the corner MVs in red. The MVs shown in red are transmitted in the bitstream for the decoder to get back the warped position.*

to utilize the framework of MV coding also for the warping parameters. The entire process of mapping, depicted in Fig. 1, can be expressed in matrix notation as,

$$\begin{bmatrix} m_0 \\ m_1 \\ m_2 \\ m_3 \\ m_4 \\ m_5 \end{bmatrix} = \begin{bmatrix} h_x & h_y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & h_x & h_y & 1 \\ -h_x & -h_y & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -h_x & -h_y & 0 \\ h_x & -h_y & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & h_x & -h_y & 0 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{bmatrix}$$

where, (h_x, h_y) denotes the centroid of the block considered. The mapped parameters $\boldsymbol{d}_{\text{cen}} = (m_0, m_1)$ contains the translational shift of the centroid of the block. As discussed earlier, $\boldsymbol{d}_{\text{cen}}$ is treated as a regular translational MV and predicted using MVs from neighboring blocks before quantization.

4.2. Transmission of Warping Parameters

The parameters $\boldsymbol{d}_{\text{left}} = (m_2, m_3)$ and $\boldsymbol{d}_{\text{right}} = (m_4, m_5)$ cannot be similarly predicted using the neighboring blocks because they depend on the sizes of current and neighboring blocks. Furthermore, the neighboring blocks may belong to objects with different motions. Therefore, we utilize the translational component (that is also available at the decoder according to Sec. 4.1) to select a subset of the neighboring blocks that have similar translational MV as the current block. The warping parameters of the subset is mapped to the current block size and predicted using median operator similar to regular translational MV. The prediction difference is quantized to quarter-sample accuracy and coded using CABAC. In the actual implementation, the MV predictor is computed first and mapped to obtain the vector $\boldsymbol{p}$ in Eq. 11 before motion estimation starts. This way, we make sure that the MV predictor used for rate-constrained motion estimation is the same as the one used in coding. With the proposed transmission scheme, the MV of a translational block can be predicted even when all the neighbors use PMM and vice versa.

Sequence	Resolution	Delta bit rate %			
		Spline		Spline+Affine	
		IPPP	IbBbB	IPPP	IbBbB
BQSquare	416x240	-10.01	-6.96	-11.72	-9.30
BasPass	416x240	-0.05	-0.51	-2.03	-0.92
BBubbles	416x240	-2.28	-1.46	-3.85	-2.76
PtyScene	832x480	-5.04	-2.88	-6.01	-3.95
BQMall	832x480	-1.19	-1.19	-2.46	-1.75
Cactus	1920x1080	0.60	0.03	-3.22	-3.83
Average		-3.01	-2.16	-4.88	-3.75

Table 1. *Delta bit rate using the proposed system. The comparison of cubic splines with H.264/AVC interpolation for quarter sample accuracy is presented under the column 'Spline'. The results for affine model with cubic spline interpolation in comparison to translational motion model is shown under the column 'Spline+Affine'.*

5. SIMULATION SETUP AND RESULTS

The reference software used for evaluating our algorithm is based on the H.264/AVC standard and is on an average 10-15 % better than the JM software due to the usage of larger blocks [7] and improved encoder optimizations. We integrated our algorithm into the reference software and measured the additional gains. We use 100 frames from each of the sequences to execute the tests. The number of reference pictures used for MC prediction is set to 4. RD optimization and deblocking filter are enabled. The codec is executed at four different quantization parameter settings, namely $QP = 22, 27, 32, 37$. The Inter mode includes translational and affine prediction modes. In the Hierarchical B GOP8 structure, the QP is cascaded according to the layer structure and the affine prediction mode is only enabled for coding the P pictures. The translational MVs in case of translational and affine prediction and the corner MVs in case of affine prediction are quantized to quarter-sample precision. Within inter blocks, a mode flag is transmitted, using the CABAC entropy coding scheme, which indicates whether the block is coded in translational or affine prediction mode.

The reference codec and the test codec are used to generate 4 RD points each and the performance improvement is measured in terms of Bjøntegaard Delta Bit Rate metric. The test results are shown in Tab. 1. There is a significant overall bit rate saving provided by the proposed approach. In particular, the cubic spline interpolation outperforms the 6-tap interpolation (even in a high precision implementation without intermediate precision loss due to rounding) for most of the sequences. For some sequences like 'Cactus', there is a slight bit rate increase due to cubic spline interpolation. However, when affine prdiction is used along with cubic splines, bit rate savings up to 11.7 % in IPPP and 9.3 % in IbBbB structures are observed for the tested sequences.

Fig. 2 depicts the relative frequency of selection of the affine prediction mode in the RD optimized mode decision. The bar graphs show the percentage of samples in a frame, averaged over 100 frames, that are predicted using the affine prediction mode. The statistics are collected for four different QPs and are shown in groups, sorted in increasing QP order in each group. It can be seen that for higher bit rates the affine prediction mode is selected more often than in the case of lower bit rates due to the extra cost of side information. For the sequence 'BBubbles', up to 22 % of the samples are predicted using the affine prediction mode.

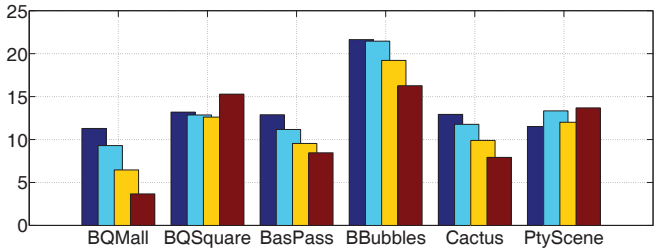

Fig. 2. *Bar graphs showing the percentage of samples in a frame predicted using affine model. Four different QPs are used for each sequence and the results are shown in groups.*

6. CONCLUSION

We presented a video coding system with improved interpolation and motion compensation capabilities. The interpolation filters in the H.264/AVC standard is replaced by a spline based framework, which provides a discrete-to-continuous mapping inside integer sample areas. This mapping is useful for estimating the translational displacement and the warping parameters. The block-wise decision whether to transmit warping parameters depends on the RD cost of the different modes.

7. REFERENCES

[1] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circ. Syst. Video Technol.*, vol. 13, pp. 560–576, Jul. 2003.

[2] G. Heising, D. Marpe, H. L. Cycon, and A. P. Petukhov, "Wavelet-based Very Low Bit-Rate Video Coding Using Image Warping and Overlapped Block Motion Compensation," *IEE Proceedings - Vision, Image and Signal Processing*, vol. 148, no. 2, pp. 93-101, Apr. 2001.

[3] Nokia MVC Codec. *ITU-T Proposal*, Feb. 2000. `http://ftp3.itu.ch/av-arch/video-site/`

[4] T. Wiegand, E. Steinbach, A. Stensrud, and B. Girod, "Multiple reference picture video coding using polynomial motion models," Proc. Visual Comm. Image Proc, vol. 3309, pp. 134–145, 1998.

[5] M. Karczewicz, J. Nieweglowski and P. Haavisto, "Video coding using motion compensation with polynomial motion vector fields," *Signal Processing: Image Commun.*, vol. 10, pp. 63-91, 1997.

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

[7] P. Chen, Y. Ye, M. Karczewicz, "Video coding using extended block sizes", *VCEG-AJ23*, San Diego, USA, 8-10 Oct. 2008.

[8] M. Unser, A. Aldroubi, M. Eden, "B-Spline Signal Processing: Part II - Efficient Design and Applications," *IEEE Transactions on Signal Processing*, vol. 41, no. 2, pp. 834-848, Feb. 1993.

[9] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, and G. J. Sullivan, "Rate-Constrained Coder Control and Comparison of Video Coding Standards", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 688-703, Jul 2003.