14th European Signal Processing Conference (EUSIPCO 2006), Florence, Italy, September 4-8, 2006, copyright by EURASIP

BLOCK MATCHING-BASED MOTION COMPENSATION WITH ARBITRARY ACCURACY USING ADAPTIVE INTERPOLATION FILTERS

Ichiro Matsuda, Kazuharu Yanagihara, Shinichi Nagashima and Susumu Itoh

Department of Electrical Engineering, Faculty of Science and Technology,
Science University of Tokyo
2641 Yamazaki, Noda-shi, Chiba 278-8510, JAPAN
Tel: +81 4 7124 1501 (ext.3722), Fax: +81 4 7124 9367
E-mail: matsuda@ee.noda.tus.ac.jp

ABSTRACT

This paper proposes a motion-compensated prediction method which can compensate precise motions using adaptive interpolation filters. In this method, plural number of non-separable 2D interpolation filters are optimized for each frame, and an appropriate combination of one of the filters and a motion vector with integer-pel accuracy is determined block-by-block. To reduce the amount of side information, coefficients of each filter are downloaded to a decoder only when the filter turns out to be effective in terms of a rate-distortion sense, or else the old filter applied to the previous frame is reused. Simulation results indicate that the proposed method provides coding gain of up to 3.0 dB in PSNR compared to the conventional motion-compensated prediction method using motion vectors with 1/2-pel accuracy.

1. INTRODUCTION

Motion-compensated prediction based on precise motion vectors, typically with 1/2-pel or 1/4-pel accuracy, is commonly used in current video coding schemes. In such schemes, predicted values corresponding to fractional-pel positions, which are located between real pels at full-pel positions in a reference image, have to be interpolated from several neighboring pels [1]. For example, the H.264/AVC standard [2] employs a two-step linear interpolation technique to realize motion-compensated prediction with 1/4-pel accuracy. In the first step, a 6-tap FIR filter is used to obtain interpolated values at 1/2-pel positions. In the second step, a simple 2-tap bilinear interpolation filter is applied to the remaining 1/4-pel positions. These two interpolation filters are time-invariant, i.e. filter coefficients are fixed for the whole sequence, and the same filters are used in horizontal and vertical directions to interpolate 2D video signals.

To decrease prediction errors within the above two-step interpolation framework, an adaptive technique based on frame-by-frame optimization of the filter coefficients was proposed [3, 4]. The optimization is carried out exclusively for the 6-tap FIR filter used in the first step under a constraint of symmetric impulse response. On the other hand, the 2-tap filter used in the second step is always fixed. Therefore the adaptive technique varies only three coefficients of the 6-tap filter for each frame. This means interpolation processes for 1/2-pel and 1/4-pel positions are unable to be optimized independently. Recently, Vatis et al. extended the adaptive interpolation technique by introducing non-separable 2D filters [5, 6]. In the proposals, positions to be interpolated are classified into five categories according to their symmetric properties and an interpolation filter is independently optimized for each category. Such a symmetric structure is also exploited to decrease the number of filter coefficients to be optimized. Though this adaptive technique can considerably reduce the amount of side information required for the frame-by-frame optimization, degrees of freedom in the filter design are still rather restricted.

This paper proposes a more flexible motion-compensated prediction method using adaptive 2D interpolation filters. In the method, a set of non-separable 2D filters are iteratively optimized for each frame, and an appropriate combination of one of the filters and a motion vector with integer-pel accuracy is determined in each macroblock composed of 16×16 pels. Since each filter is not associated with fractional parts of motion vectors and is optimized independently without a constraint of the symmetric structure of filter coefficients, there is virtually no limitation on accuracy of motions to be compensated. Moreover, effectiveness of the optimized filter is compared with that of the old filter which has been used in the previous frame. If the latter turns out to be more effective in terms of a rate-distortion sense, it is reused and consequently no coefficient of the filter is downloaded. This renewal strategy of filters is useful for preventing the side information from increasing unnecessarily.

2. MOTION-COMPENSATED PREDICTION USING ADAPTIVE INTERPOLATION FILTERS

The proposed method detects a motion vector $\boldsymbol{v}$ in each macroblock composed of 16×16 pels using the block matching algorithm. Both horizontal and vertical components of the motion vector are integer values of $[-15, +15]$. In return for such a seeming limitation on motion vector accuracy, the method allows a variety of interpolation process using M kinds of non-separable 2D filters [7]. Predicted values in each macroblock are calculated by applying one of the interpolation filters to a motion-compensated reference image as shown in Figure 1. When a pel $\boldsymbol{p}$ belongs to a macroblock where the m-th filter ($m = 1, 2, \ldots, M$) is selected, a predicted value $\hat{s}(\boldsymbol{p})$ is given by the following 2D convolution:

$$\hat{s}(\boldsymbol{p}) = \sum_{k=1}^{K} a_m(k) \cdot s'(\boldsymbol{p} + \boldsymbol{v} + \boldsymbol{q}_k), \qquad (1)$$

where $s'(\boldsymbol{p})$ indicates the reference image (i.e. a reconstructed image of the previous frame) and $a_m(k)$ is the k-th coefficient of the m-th filter with a 2D filter mask of size K. Note that a shape of the mask can be other than rectangle and is defined

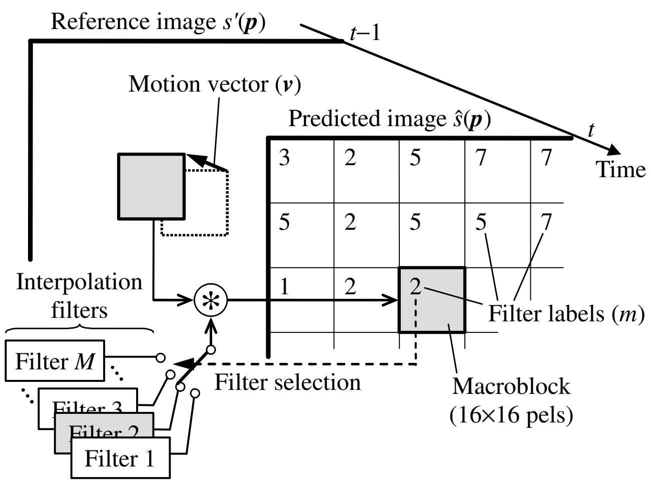


Figure 1: Illustration of the proposed motion-compensated prediction method.

by a series of 2D vectors $\{\boldsymbol{q}_k \,|\, k = 1,2,\ldots,K\}$ in this paper (see Figure 4). If a composite vector $\boldsymbol{p}+\boldsymbol{v}+\boldsymbol{q}_k$ points to outside of the reference image, a value of $s'(\boldsymbol{p}+\boldsymbol{v}+\boldsymbol{q}_k)$ in Eq.(1) is taken from the nearest pel on the border of the image.

In the existing methods [3]–[6], an interpolation filter used in each macroblock is directly connected with a motion vector detected in the macroblock. In other words, fractional parts of motion vectors indicate the interpolation filter applied to the macroblock. Hence, the number of available filters relies on accuracy of motion vectors and is generally unchangeable. Moreover, a structure of each filter depends on positions to be interpolated. When a motion vector points to full-pel positions, for example, no filter is used for the prediction. It may limit the effect of noise reduction obtained by spatial low-pass filtering [1]. On the other hand, our method can utilize any number of filters and takes full advantage of the effect of noise reduction even in the case of no motion.

3. OPTIMIZATION PROCEDURES

In order to generate a predicted image at the decoder side, the proposed method has to encode the following parameters as side information.

- Motion vector ($\boldsymbol{v}$) with integer-pel accuracy detected in each macroblock.
- Filter label (m) which specifies the interpolation filter applied to the macroblock.
- Coefficients $\{a_m(k) \,|\, k = 1,2,\ldots,K\}$ of newly designed filters ($m \in \{1,2,\ldots,M\}$) in each frame.

These parameters are iteratively optimized for each frame so that a value of the following cost function J can be a minimum:

$$J = D + \lambda \cdot B_{side}, \tag{2}$$

where D is distortion of the predicted image measured by Sum of Squared Errors (SSE), B_{side} is the number of bits required for the above side information and λ is a constant called the Lagrangian multiplier [8], respectively. Concrete procedures for the optimization are as follows:

(1) Initial motion vectors $\boldsymbol{v}^{(0)}$s with integer-pel accuracy are determined by the block matching algorithm. Any interpolation filter is not applied yet at this stage.

(2) Initial interpolation filters $\{a_m^{(0)}(k) \,|\, k = 1,2,\ldots,K\}$ ($m = 1,2,\ldots,M$), which have been used in the previous frame, are tested for each macroblock. Then the label m corresponding to the best matched filter which minimizes the cost function J is assigned to the macroblock. When the initial filters are unavailable, that is, when the previous frame is intra-coded, provisional labeling based on M-level quantization of the SSE obtained by block matching in the procedure (1) is performed [1].

(3) The optimum filter is designed for the area composed of macroblocks to which the same label m is assigned. In practice, it is difficult to obtain the optimum filter coefficients which exactly minimize a value of the cost function J. Therefore we temporarily neglect fluctuations of J caused by quantization and coding of the filter coefficients. As a result, new filter coefficients $\{a_m^{(n)}(k) \,|\, k = 1,2,\ldots,K\}$, which minimize only the distortion D in Eq.(2), are easily obtained by solving linear simultaneous equations known as the Wiener-Hopf equations [3] [2].

(4) To check the validity of the new filter, an actual value of the cost function J is calculated. If the value is worse than before, the filter is discarded and all the filter coefficients are restored to the preceding state:

$$a_m^{(n)}(k) \leftarrow a_m^{(n-1)}(k) \quad \text{for} \quad k = 1,2,\ldots,K. \tag{3}$$

Then, the filter is again compared with the initial one. Obviously, the initial filter never minimizes the distortion D, however it has an advantage of no additional information on its coefficients $a_m^{(0)}(k)$s. Accordingly, if the initial filter turns out to be better in terms of the cost function J, it is substituted for the current filter:

$$a_m^{(n)}(k) \leftarrow a_m^{(0)}(k) \quad \text{for} \quad k = 1,2,\ldots,K. \tag{4}$$

(5) A new motion vector $\boldsymbol{v}^{(n)}$ is searched within an area which is composed of 3×3 pels and is centered at its original position ($\boldsymbol{v}^{(n-1)}$). At the same time, M kinds of interpolation filters are tested at every search point. Finally, the best combination of the motion vector $\boldsymbol{v}^{(n)}$ and the filter label m is selected from $9 \times M$ candidates in each macroblock.

(6) Procedures (3), (4) and (5) are iteratively carried out while increasing a value of the loop counter n by 1 for each iteration. In the current implementation, the iteration terminates when the cost function J can no longer decrease. However, setting the maximum number of iterations would be helpful for reducing computational complexity of the encoder.

4. CODING OF THE PARAMETERS

This section describes coding of the above mentioned parameters.

[1] There is no particular reason to use the SSE for this process. Our experiments, however, suggest that the final result is not so sensitive to the provisional labeling methods.

[2] Though a wrong filter can be obtained when uniqueness of the solution is not guaranteed, stability of the proposed method would be kept because such a filter is discarded in the procedure (4).

4.1 Motion vectors

Motion vectors are differentially encoded using an H.263-like median prediction method [9]. However, a variable-length code (VLC) table for the differential motion vector is borrowed from the old H.261 standard [10] as an example, because motion vectors have integer-pel accuracy in our method as well as the standard.

4.2 Filter labels

In general, values of the label m in adjacent macroblocks correlate to each other. Nevertheless, simple differential coding of these values is not effective because a value of the label m is assigned to each interpolation filter regardless of similarity among filters. Therefore we introduce a kind of transformation algorithm called the move-to-front (MTF) method [11]. The MTF method expresses a symbol by its position in a look-up table whose elements are constantly rearranged so that the most probable symbol can be placed in the front. In our method, three labels of top, left and top-right macroblocks are moved to the front positions in this order (if two of them are the same, it is placed at the first position based on a majority rule). Figure 2 illustrates this process by using an example. In the figure, a series of the filter labels $m = 5, 2, 5, 5, 7 \cdots$ are transformed into a sequence of $2, 1, 0, 1, 0 \cdots$. This transformation is reversible and the transformed values tend to be small when the original values are correlated. Therefore, instead of encoding the filter labels directly, we efficiently encode the transformed values using an optimum Huffman code table.

4.3 Filter coefficients

At first, the proposed method encodes a 1-bit flag which shows whether the filter coefficients will be downloaded or not for each filter ($m = 1, 2, \ldots, M$). Then values of the coefficients to be downloaded $\{a_m^{(n)}(k) \,|\, k = 1, 2, \ldots, K\}$ are quantized with step size Δa and differences from the initial values $\{a_m^{(n)}(k) - a_m^{(0)}(k)\}$ are encoded using VLCs. We prepare 8 kinds of VLC tables for this differential coding and the optimum one is selected for each filter.

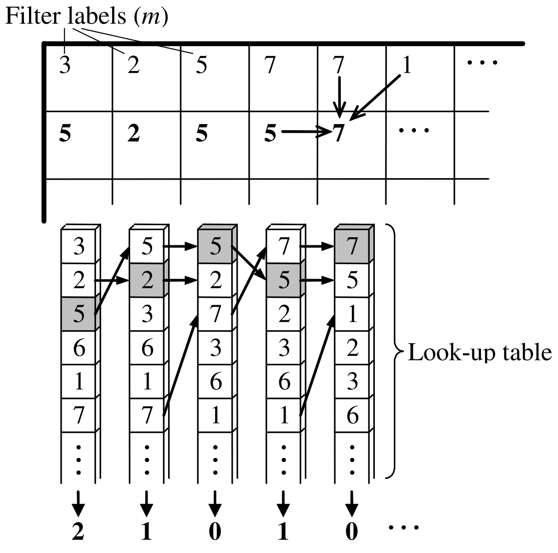


Figure 2: Coding of filter labels (m) using the move-to-front method.

5. PERFORMANCE EVALUATION

The proposed method is incorporated into the H.263-based DCT coding algorithm [9] to investigate its effectiveness. CIF-sized monochrome sequences of 1 second long (30 frames at 30 Hz) are encoded with a GOP structure of IPPP$\cdots$. Motion-compensated prediction is applied to all of the macroblocks in P-pictures (i.e. no intra-mode is used). Quantization step size Q for DCT coefficients of prediction residuals is fixed for the whole sequence and the Lagrangian multiplier λ in Eq.(2) is given by an empirical equation reported in [8]:

$$\lambda = 0.85 \cdot (Q/2)^2. \qquad (5)$$

In the following results, Δbitrate represents average bitrate reduction from the conventional method which utilizes motion compensation with 1/2-pel accuracy and bilinear interpolation. The detailed definition of Δbitrate is described in [12].

Figure 3 shows the relationship between Δbitrate and filter size. We tested two types of 2D filter masks: square-shaped and diamond-shaped masks, while changing the filter size, namely, the number of filter coefficients (K). In both

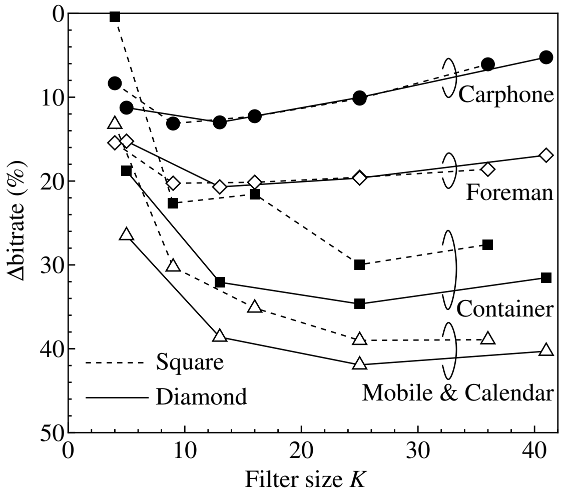
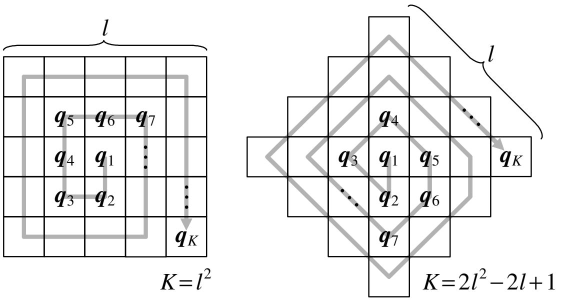


Figure 3: Δbitrate vs. filter size (K).



(a) Square

(b) Diamond

Figure 4: 2D filter masks.

masks, filter coefficients are disposed in spiral order and a value of K is expressed as a function of side length l as illustrated in Figure 4. In this experiment, the accuracy of filter coefficients and the number of interpolation filters are fixed to $\Delta a = 2^{-6}$ and $M = 16$, respectively. We can see that the use of too large filter masks deteriorates coding performance due to a lot of side information on the filter coefficients and that $K = 25$ is a reasonable choice for most sequences. Moreover, the diamond-shaped mask is superior to the square-shaped one for the sequences 'Container' and 'Mobile & Calendar'. Therefore we utilize the diamond-shaped filter mask with size of $K = 25$ hereafter.

We conduct similar experiments with respect to other coding parameters as well. Figures 5 and 6 indicate influence of the accuracy of filter coefficients (Δa) and the number of interpolation filters (M) on the coding performance respectively. It is observed in Figure 5 that the best results are obtained when Δa is set to 2^{-6} except for the 'Container' sequence. By setting a value of Δa to a power of two, the filtering process can be implemented using fixed-point arithmetic which enables fast computation. Accordingly the value of $\Delta a = 2^{-6}$ seems to be suitable for practical applications. On the other hand, Figure 6 shows that increase of the number of interpolation filters generally leads to better coding performance, however the gain is nearly saturated when M becomes larger than 16. As a matter of fact, the final number of interpolation filters used in the coding process tends to be smaller than the given number M especially at low coding rates, because the optimization based on the cost function J automatically removes ineffective filters. In the procedure (5) mentioned in Section 3, for example, if some filter label m' is not selected in any macroblock, the interpolation filter corresponding to the label m' is removed. In this way, the proposed method adaptively changes the actual number of interpolation filters according to coding rates and/or properties of a sequence.

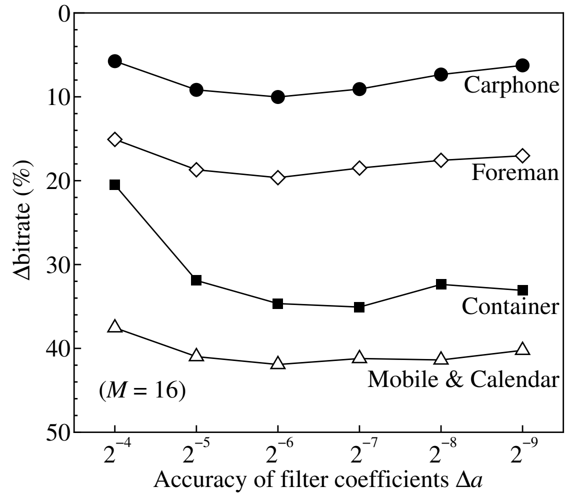


Figure 5: Δbitrate vs. accuracy of filter coefficients (Δa).

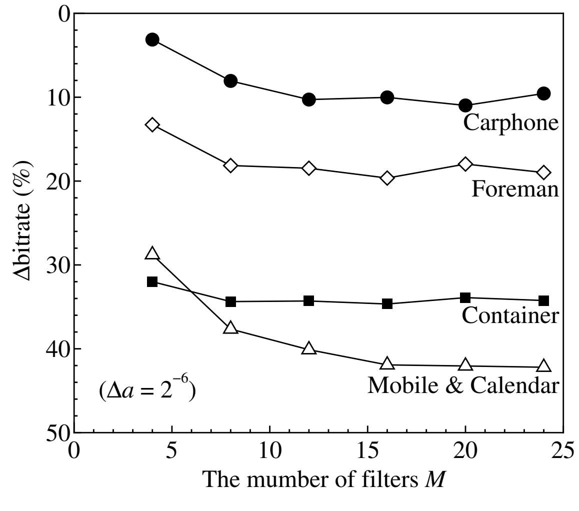


Figure 6: Δbitrate vs. the number of interpolation filters (M).

Figure 7 indicates rate-distortion curves provided by three kinds of motion-compensated prediction methods. In this figure, '1/4-pel MC' and '1/2-pel MC' represent the conventional methods using motion vectors with 1/4-pel and 1/2-pel accuracy. In the case of '1/4-pel MC', the two-step interpolation with the 6-tap filter is applied in the same way as the H.264/AVC standard. With regard to the proposed method, $M = 16$, $\Delta a = 2^{-6}$ and $K = 25$ with the diamond-shaped filter mask are adopted. It is demonstrated that the proposed method outperforms the two conventional methods for all the tested sequences. When the proposed method is compared to the '1/4-pel MC' and '1/2-pel MC', the coding gain is up to 0.8 dB and 3.0 dB in PSNR respectively.

6. CONCLUSIONS

This paper has proposed a novel motion-compensated prediction method using adaptive 2D interpolation filters. Instead of increasing accuracy of motion vectors, we introduce adaptive selection of the non-separable 2D interpolation filters in each macroblock. Not only motion vectors but also filter coefficients and filter assignment are iteratively optimized for each frame and then encoded as side information. This optimization process is carried out so that the rate-distortion-based cost function can be a minimum. In addition, coefficients of each filter are downloaded only when the cost function can decrease as a result of applying the filter. Simulation results indicate that the proposed method attains better coding performance than the conventional methods using motion vectors with fractional-pel accuracy.

Since the proposed method is based on frame-by-frame optimization of several kinds of parameters, the encoding process requires a relatively large amount of computation at the encoder side. For instance, our prototype encoder takes a few seconds/frame on a computer with a 3.6 GHz Intel Xeon processor. However, decoding speed is fast enough for practical video applications because the optimization process is not necessary at the decoder side. Fast implementation of the encoder as well as extensions of the proposed method to bidirectional prediction, multi-frame prediction and/or variable block-size motion estimation will be our future work.

REFERENCES

[1] B. Girod, "Motion-Compensating Prediction with Fractional-Pel Accuracy," IEEE Trans. on Communications, Vol. 41, No. 4, pp.604–612, Apr. 1993.

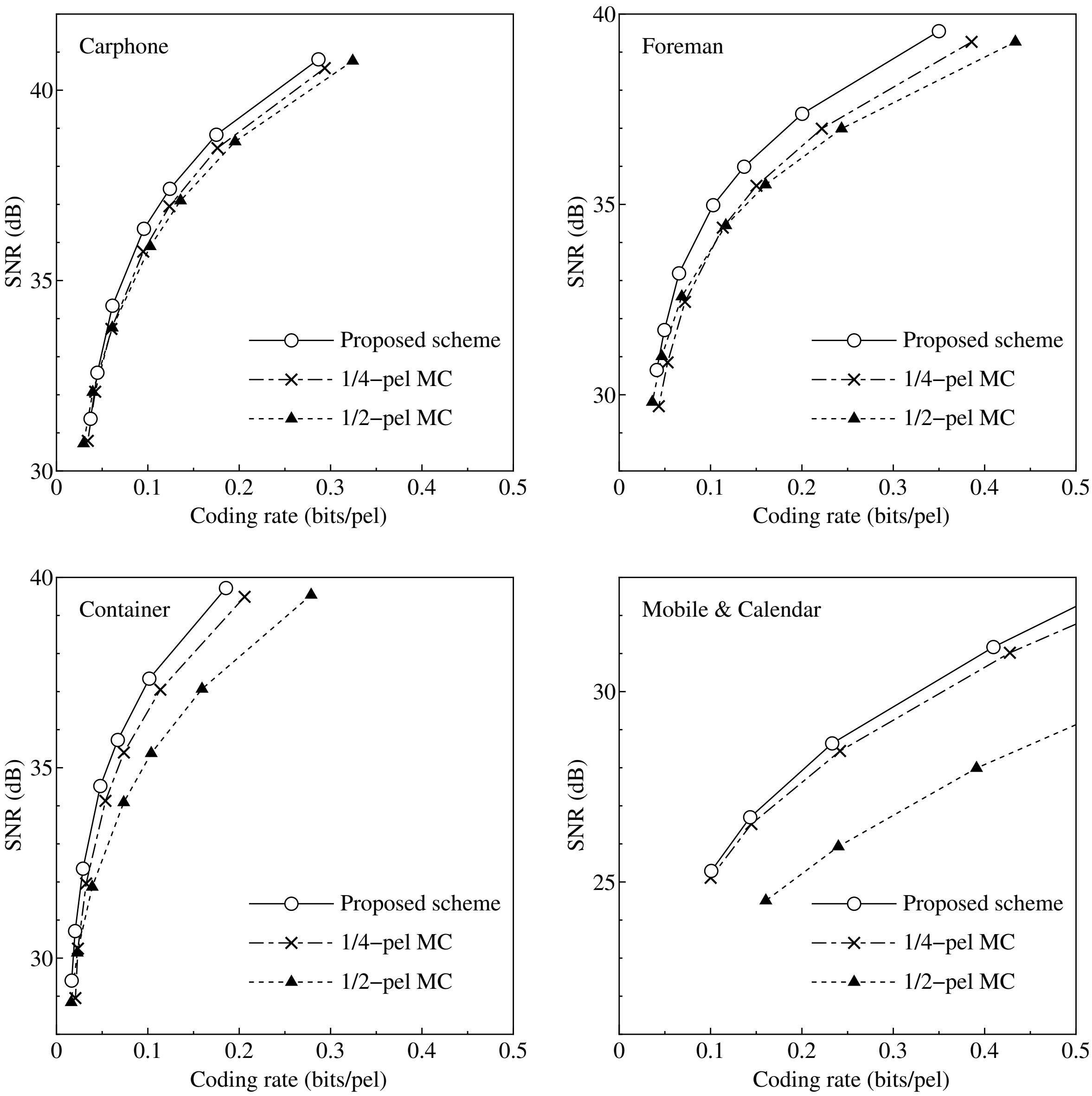


Figure 7: Coding performance.

[2] ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC, “Advanced Video Coding for Generic Audiovisual Services,” 2003.

[3] T. Wedi, “Adaptive Interpolation Filter for Motion Compensated Hybrid Video Coding,” Proc. of Picture Coding Symposium (PCS 2001), pp.49–52, Apr. 2001.

[4] T. Wedi, “Adaptive Interpolation Filter for Motion Compensated Prediction,” Proc. of 2002 IEEE International Conference on Image Processing (ICIP 2002), Vol. II, pp.509–512, Sep. 2002.

[5] Y. Vatis, B. Edler, D. T. Nguyen and J. Ostermann, “Motion-and Aliasing-compensated Prediction using a Two-Dimensional Non-Separable Adaptive Wiener Interpolation Filter,” Proc. of 2005 IEEE International Conference on Image Processing (ICIP 2005), pp.894–897, Sep. 2005.

[6] Y. Vatis, B. Edler, I. Wassermann, D. T. Nguyen and J. Ostermann, “Coding of Coefficients of Two-Dimensional Non-Separable Adaptive Wiener Interpolation Filter,” Proc. of SPIE: Visual Communication and Image Processing (VCIP 2005), Vol.5960, pp.623–631, July 2005.

[7] H. Shirasawa, I. Matsuda and S. Itoh, “Motion Compensation with Fractional-Pel Accuracy Using Adaptive Interpolation,” Proc. of 1st Forum on Information Technology (FIT 2002), No.J-74, pp.349–350, Nov. 2002 (in Japanese).

[8] G. J. Sullivan and T. Wiegand, “Rate-Distortion Optimization for Video Compression,” IEEE Signal Processing Magazine, pp.74–90, Nov. 1998.

[9] ITU-T Rec. H.263, “Video Coding for Low Bitrate Communication,” 1996.

[10] ITU-T Rec. H.261, “Video Codec for Audiovisual Services at $p\times64$ kbit/s,” 1993.

[11] J. L. Bentley, D. D. Sleator, R. E. Tarjan and V. K. Wei, “A Locally Adaptive Data Compression Scheme,” Communications of ACM, Vol.29, No.4, pp.320–330, Apr. 1986.

[12] G. Bjontegaard, “Calculation of Average PSNR Differences Between RD-Curves,” ITU-T SG16 Doc. VCEG-M33, Apr. 2001.