INTERNATIONAL ORGANIZATION FOR STANDARDIZATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC1/SC29/WG11
CODING OF MOVING PICTURES AND ASSOCIATED AUDIO

ISO/IEC JTC1/SC29/WG11
MPEG04/M10569/S03
Munich, March 2004

Title: Scalable Extension of H.264/AVC
Status: Proposal
Source: Fraunhofer Institute for Telecommunications – Heinrich Hertz Institute
Authors: Heiko Schwarz, Detlev Marpe, and Thomas Wiegand

Abstract

We propose a scalable extension of the H.264/AVC video coding standard. To achieve an efficient scalable bit-stream representation of a video sequence, the temporal dependencies between pictures are exploited by using an open-loop subband approach. The related temporal analysis-synthesis filter bank structure is generalized to facilitate an adaptive block-based choice between the motion-compensated lifting representations of the Haar filter (uni-directional prediction) and the 5/3 filter (bi- prediction), both coupled with multiple-reference frame capabilities. Furthermore, an intra mode can be chosen on a block basis to efficiently represent blocks that cannot be reasonably predicted using motion compensation. As a remarkable feature of our approach, most components of H.264/AVC are used as specified in the standard, while only a few have been adjusted to the motion-compensated temporal filtering structure.

1 Introduction

In recent years, several efficient video codecs using motion-compensated temporal filtering have been presented [2][3][4][5]. The main reason for the recent advances in temporal subband coding is the utilization of the lifting representation [6] of a filter bank in the temporal direction. A two-channel decomposition can be achieved by a sequence of prediction and update steps. Since the lifting structure is invertible without requiring invertible prediction and update steps, motion-compensated prediction using any possible motion model can be incorporated into the prediction and update steps.

By using the highly efficient motion model of the H.264/AVC standard [1] in connection with an adaptive switching between the Haar and the 5/3 spline wavelet on a block basis, both the prediction and the update step are similar to the motion-compensated prediction of B slices as specified in the H.264/AVC standard. Furthermore, the open-loop structure of a temporal subband representation offers the possibility to efficiently incorporate temporal and quality (SNR) scalability. Spatial scalability can be added by adapting the spatial scalability concept as it is found in the video coding standards H.262/MPEG-2 Visual [7], H.263 [8], or MPEG-4 Visual [9] to the subband structure obtained by motion-compensated temporal filtering. Motivated by these facts, we have investigated the possibility of a simple but yet efficient scalable extension of H.264/AVC.

2 Temporal Decomposition and its Integration into H.264/AVC

In this section, we briefly review the lifting scheme and explain how it is applied to H.264/AVC video coding.

2.1 Review of Motion-Compensated Temporal Filtering (MCTF)

The generic lifting scheme consists of three steps: polyphase decomposition, prediction, and update. Figure 1 illustrates the lifting representation of an analysis-synthesis filter bank.

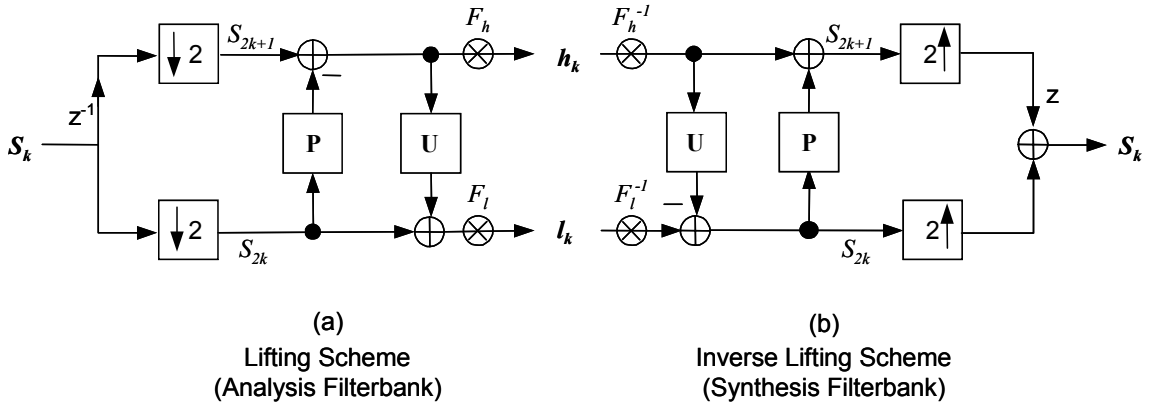

Figure 1: Lifting representation of an analysis-synthesis filter bank.

At the analysis side (a), the odd samples $s[2k+1]$ of a given signal s are predicted by a linear combination of the even samples $s[2k]$ using a prediction operator $\mathbf{P}(s[2k+1])$ and a high pass signal $h[k]$ is formed by the prediction residuals. A corresponding low-pass signal $l[k]$ is obtained by adding a linear combination of the prediction residuals $h[k]$ to the even samples $s[2k]$ of the input signal s using the update operator $\mathbf{U}(s[2k])$:

$$h[k] = s[2k+1] - \mathbf{P}(s[2k+1]) \qquad \text{with} \quad \mathbf{P}(s[2k+1]) = \sum_i p_i \, s[2k+2i]$$

$$l[k] = s[2k] \quad + \mathbf{U}(s[2k]) \qquad \text{with} \quad \mathbf{U}(s[2k]) \quad = \sum_i u_i \, h[k+i]$$

Since both the prediction and the update step are fully invertible, the corresponding transform can be interpreted as critically sampled perfect reconstruction filter bank. The synthesis filter bank simply consists of the application of the prediction and update operators in reverse order with the inverted signs in the summation process followed by the reconstruction process using the even and add polyphase components. For a normalization of the low- and high-pass components, appropriately chosen scaling factors F_l and F_h are applied, respectively. In practice, these scaling factors don't need to be applied during the decomposition and reconstruction process, but can be incorporated when selecting the quantisation step sizes during encoding.

Let $s[\mathbf{x}, k]$ be a video signal with the spatial coordinate $\mathbf{x} = (x, y)^T$ and the temporal coordinate k. The prediction and update operators for the temporal decomposition using the lifting representation of the Haar wavelet are given by

$$\mathbf{P}_{Haar}(s[\mathbf{x}, 2k+1]) = \quad s[\mathbf{x}, 2k]$$

$$\mathbf{U}_{Haar}(s[\mathbf{x}, 2k]) \quad = \frac{1}{2} h[\mathbf{x}, k]$$

For the 5/3 transform, the prediction and update operators are given by

$$\mathbf{P}_{5/3}(s[\mathbf{x}, 2k+1]) = \frac{1}{2}(s[\mathbf{x}, 2k] + s[\mathbf{x}, 2k+2])$$

$$\mathbf{U}_{5/3}(s[\mathbf{x}, 2k]) \quad = \frac{1}{4}(h[\mathbf{x}, \ k] + h[\mathbf{x}, \ k-1])$$

The extension to motion-compensated temporal filtering is realized by modifying the prediction and update operators as follows

$$\mathbf{P}_{Haar}(s[\mathbf{x}, 2k+1]) = \quad s[\mathbf{x}+\mathbf{m}_{P0}, 2k-2r_{P0}]$$

$$\mathbf{U}_{Haar}(s[\mathbf{x}, 2k]) \quad = \frac{1}{2} h[\mathbf{x}+\mathbf{m}_{U0},\ k+\ r_{U0}]$$

$$\mathbf{P}_{5/3}(s[\mathbf{x}, 2k+1]) = \frac{1}{2}(s[\mathbf{x}+\mathbf{m}_{P0}, 2k-2r_{P0}] + s[\mathbf{x}+\mathbf{m}_{P1}, 2k+2+2r_{P1}])$$

$$\mathbf{U}_{5/3}(s[\mathbf{x}, 2k]) \quad = \frac{1}{4}(h[\mathbf{x}+\mathbf{m}_{U0},\ k+\ r_{U0}] + h[\mathbf{x}+\mathbf{m}_{U1},\ k-1-\ r_{U1}])$$

where the reference indices $r \geq 0$ allow a general frame-adaptive motion-compensated filtering as proposed in [5]. The motion vectors **m** are not restricted to sample-accurate displacements. In case of sub-sample accurate motion vectors, the term $s[\mathbf{x}+\mathbf{m}, k]$ has to be interpreted as a spatially interpolated value.

As can be seen from the above equations, both the prediction and update operators for the motion-compensated filtering using the lifting representation of the Haar wavelet are equivalent to uni-directional motion-compensated prediction. For the 5/3 spline wavelet, the prediction and update operators specify bi-directional motion-compensated prediction.

Since bi-directional motion-compensated prediction generally reduces the energy of the prediction residual but increases the motion vector rate in comparison to uni-directional prediction, it is desirable to switch dynamically between uni- and bi-directional prediction, and thus between the lifting representation of the Haar and the 5/3 spline wavelet.

2.2 Integration into H.264/AVC

To represent the motion fields, or more accurately the prediction data arrays M_P and M_U, for the prediction and update operators, we use the existing syntax for B slices in H.264/AVC [1]. As a slight modification, the direct macroblock and sub-macroblock mode are redefined. They specify that the corresponding macroblock or sub-macroblock is bi-directionally predicted, that the reference indices are equal to zero, and that the list 0 (backward) and list 1 (forward) motion vectors for the 16×16 or 8×8 block are given by the corresponding spatial motion vector predictors.

Furthermore, we also incorporate an intra mode. For the intra macroblock mode, the following prediction and update operators are used

$$\mathbf{P}_{Intra}(s[\mathbf{x}, 2k+1]) = 0$$

$$\mathbf{U}_{Intra}(s[\mathbf{x}, 2k]) \quad = 0$$

An intra macroblock mode in a prediction data array M_P specifies that in the corresponding prediction step at the analysis side, the macroblock samples of the original low-pass signal are placed into the high-pass picture. For the update step, an intra macroblock mode in a prediction data array M_U indicates that the update of the low-pass signal is skipped for the corresponding macroblock. It should be noted that motion vectors of the prediction data array M_U used in the update steps could reference an area in a high-pass picture that partially or fully covers an intra macroblock. Since the intra macroblocks in the high-pass pictures should not be used for updating the low-pass pictures, all sample values of the intra macroblocks are set to zero for the usage in the update process (cp. sec. 2.3.4). Our simulation results show that the incorporation of the intra macroblock mode increases coding efficiency, especially for sequences with strong local motion.

In general, a prediction data array M_P or M_U specifies the prediction methods and the associated parameters as follows (cp. H.264/AVC [1]).

- For each macroblock (a macroblock covers an area of 16x16 luma samples), a macroblock mode is specified, which can be equal to B_Direct_16x16, B_L0_16x16, B_L1_16x16, B_Bi_16x16, B_L0_L0_16x8, B_L0_L1_16x8, B_L0_Bi_16x8, B_L1_L0_16x8, B_L1_L1_16x8, B_L1_Bi_16x8, B_Bi_L0_16x8, B_Bi_L1_16x8, B_Bi_Bi_16x8, B_L0_L0_8x16, B_L0_L1_8x16, B_L0_Bi_8x16, B_L1_L0_8x16, B_L1_L1_8x16, B_L1_Bi_8x16, B_Bi_L0_8x16, B_Bi_L1_8x16, B_Bi_Bi_8x16, B_8x8, or INTRA.

- In case the macroblock mode is equal to B_8x8, for each sub-macroblock (a sub-macroblock covers an area of 8x8 luma samples), a corresponding sub-macroblock mode is specified, which can be equal to B_Direct_8x8, B_L0_8x8, B_L1_8x8, B_Bi_8x8, B_L0_8x4, B_L1_8x4, B_Bi_8x4, B_L0_4x8, B_L1_4x8, B_Bi_4x8, B_L0_4x4, B_L1_4x4, or B_Bi_4x4.
- In case the macroblock mode is not equal to INTRA,
  - for each macroblock partition, one (in case of uni-directional prediction) or two (in case of bi-prediction) reference indices are specified,
  - for each macroblock or sub-macroblock partition, one (in case of uni-directional prediction) or two (in case of bi-prediction) motion vectors with quarter-sample accuracy are specified.

Using the described syntax for specifying the prediction data arrays M_P and M_U, the formation of the prediction and update pictures $P(\, s[\mathbf{x}, 2k+1] \,)$ and $U(\, s[\mathbf{x}, 2k] \,)$ is nearly identical to the B slice reconstruction process (prior to the deblocking filter) as specified in H.264/AVC [1]. The following main differences can be identified (cp. sec. 2.3.4):

- The derivation process for the motion vectors and reference indices used in the direct macroblock and sub-macroblock mode is simplified.
- The INTRA mode reconstruction process is simplified, since all samples are set to zero.
- The reconstruction process for motion-compensated prediction modes is simplified, since no prediction residual is added.

The deblocking filter is only applied to the low-pass pictures that are reconstructed in the prediction steps at the decoder side (cp. sec. 2.3.2 and 2.3.6).

In order to reduce the bit-rate needed for transmitting the prediction data arrays, the prediction data arrays M_U used in the update steps are neither estimated nor coded. Instead, they are derived from the set of prediction data arrays M_P used in the prediction steps of the same decomposition / composition stage. The process for deriving the prediction data arrays M_U is designed in a way that the derived prediction data arrays M_U still represent block-wise motion compatible with the B slice syntax of H.264/AVC [1].

In principle, the derivation process works as follows. Initially, the prediction data arrays M_U are divided into 4×4 blocks and for each block, a variable N_{Covered} is set to zero. Then, for each motion vector $\mathbf{m}$ of the prediction data arrays M_P, the 4×4 blocks that are at least partially covered by the area used for motion-compensated prediction of the corresponding partition are identified. If the motion vector that is already assigned to such a covered block is equal to $-\mathbf{m}$, the value of the associated variable N_{Covered} is increased by the number of newly covered samples. Otherwise, if the number of covered samples is greater than N_{Covered} for an identified block, the motion vector of the corresponding block is set equal to $-\mathbf{m}$. After processing all motion vectors of the prediction data arrays M_P, the macroblock modes and, if appropriate, the sub-macroblocks modes of the prediction data arrays M_U are determined based on the motion vectors and the variables N_{Covered} of the corresponding 4×4 blocks. Due to the limitations imposed by the H.264/AVC syntax the actual process for the derivation of the prediction data arrays M_U that are used in the update steps is a bit more complicated, but it still follows the described principle. A detailed description of the derivation process is given in sec. 2.3.5.

Following the motion-compensated filtering process as described in sec. 2.1, a group of N_0 input pictures is decomposed into groups of $(N_0 + 1) / 2$ low-pass pictures and $N_0 / 2$ high-pass pictures both with half the temporal resolution as the group of input pictures. As depicted in Figure 2(a), each generated low-pass picture (with index i) shares the coordinate system with the corresponding even-numbered input picture (with index $2i$), and each generated high-pass pictures (with index i) shares the coordinate system with the corresponding odd-numbered input picture (with index $2i+1$). In case the update steps of the decomposition process are skipped, the sequence of generated low-pass pictures is identical to the sequence of even-numbered input pictures. In Figure 2, the correspondences between the input pictures and the low- and high-pass pictures are illustrated with dashed lines; the arrows indicate which pictures can be used for motion-compensated prediction (for clarity, only the simple case, in which the reference index lists have been constructed without reordering and the number of active entries was set to 1 (cp. sec. 2.3.3), is illustrated).

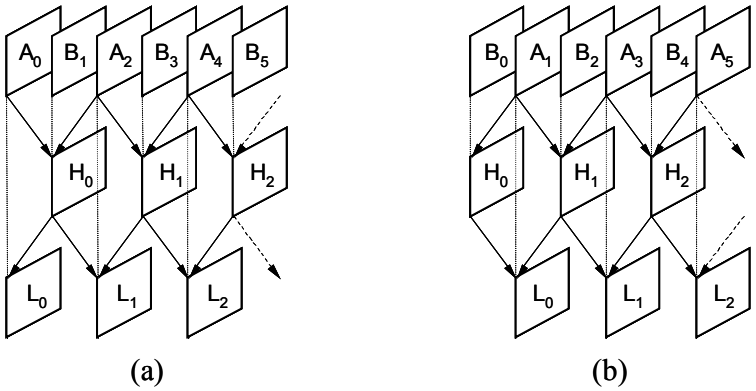

Figure 2: Temporal decomposition of a sequence of input pictures into sequences of low- and high-pass pictures both with half the temporal resolution of the sequence of input picture: (a) temporal decomposition using a correspondence between low-pass pictures and even-numbered input pictures as well as between high-pass pictures and odd-numbered input pictures, (b) temporal decomposition using a correspondence between low-pass pictures and odd-numbered input pictures as well as between high-pass pictures and even-numbered input pictures.

It is easy to see that the decomposition structure can be slightly changed, such that a correspondence between the low-pass pictures and the odd-numbered input pictures as well as between the high-pass pictures and the even-numbered input pictures is established (see Figure 2(b)). The decomposition can be further generalized as follows. A group of N_0 input pictures is partitioned into a set of N_A ($0 < N_A < N_0$) input pictures and a set of $N_B = N_0 - N_A$ input pictures. The pictures of the first set are labelled as pictures *A*, and the pictures of the second set are labelled as pictures *B*. This partitioning can generally be specified using a bit string of N_0 bits. The decomposition is performed in a way that correspondences between the generated low-pass pictures and the input pictures *A* as well as between the generated high-pass pictures and the input pictures *B* are obtained. Note, that for generating the high-pass pictures in the prediction step, only the input pictures *A* can be used as reference pictures for predicting an input picture *B*. Using this generalized decomposition scheme, it is for instance possible to design the decomposition process in a way that the ratio of the temporal resolutions of the generated low-pass sequence and the input sequence is $1/n$ ($n > 1$). In Figure 3(a), a decomposition scheme for a temporal resolution ratio of 1/3 is illustrated (cp. [2]). More general temporal resolution ratios of m/n ($m > 0$, $n > m$) can only be realized on average, since the temporal distance between two successive low-pass pictures is always a multiple of the temporal distance between successive input pictures (see Figure 3(b) for a temporal resolution ratio of 2/3).

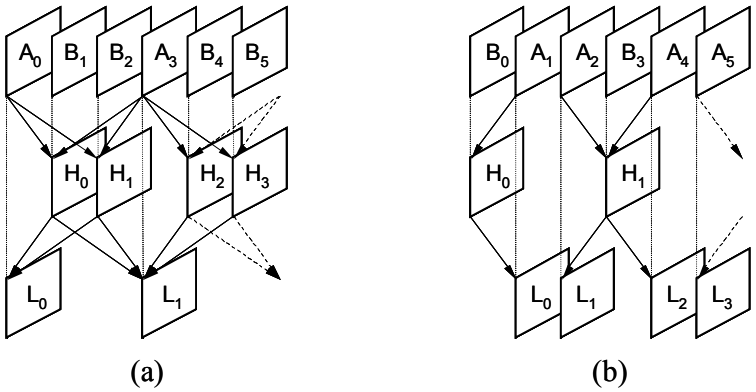

Figure 3: Temporal decomposition of a sequence of input pictures into a sequence of low-pass pictures and a sequence of high-pass pictures: (a) the temporal resolution of the sequence of low-pass pictures is 1/3 of the temporal resolution of the input sequence, (b) the temporal resolution of the sequence of low-pass pictures is on average 2/3 of the temporal resolution of the input sequence.

For groups of $N_0 > 2$ pictures it is in general advantageous to apply a multi-channel decomposition instead of a two-channel decomposition. Therefore, the presented two-channel decomposition is iteratively applied to the set the low-pass pictures until a single low-pass picture is obtained or a given number of decomposition stages has been performed. By applying *n* decomposition stages, up to *n* levels of temporal scalability can be realized. In Figure 4, the decomposition of a group of 12 pictures is illustrated; in this example 3 levels of temporal scalability with temporal resolution ratios of 1/2, 1/4, and 1/12 are provided. A detailed description of the decomposition and reconstruction process for groups of N_0 pictures is given in sec. 2.3.

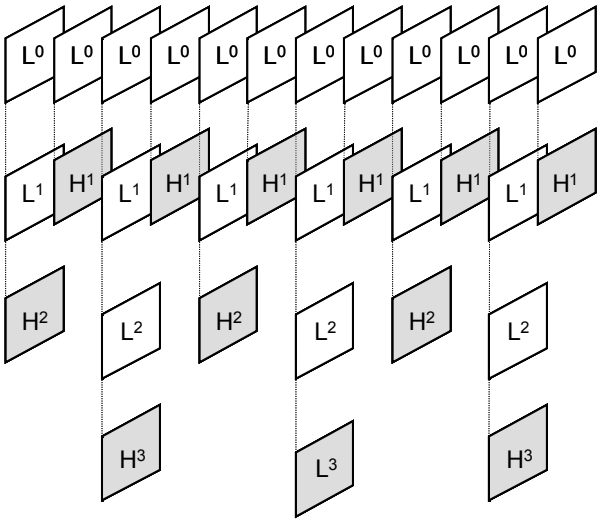

Figure 4: Temporal decomposition of a group of 12 pictures providing 3 levels of temporal scalability with temporal resolution ratios of 1/2, 1/4, and 1/12.

2.3 Detailed Description of the Decomposition and Reconstruction Process

The decomposition of a given group of N_0 original (low-pass) pictures $\{ L_0[0], \ldots, L_0[N_0-1] \}$ into an ordered set of low-pass pictures $\{ L_n[0], \ldots, L_n[N_n-1] \}$ with reduced temporal resolution and various ordered sets of high-pass pictures $\{ H_1[0], \ldots, H_1[N_0-N_1-1] \}$, $\{ H_2[0], \ldots, H_2[N_1-N_2-1] \}$, …, $\{ H_n[0], \ldots, H_n[N_{n-1}-N_n-1] \}$ is realized by the following algorithm:

- The number of decomposition stages n is initially set to zero.
- While the number of low-pass pictures N_n is greater than 1 and the number of already performed decomposition stages n is less than a possibly given number of maximum decomposition stages, the following applies.
  - The encoder control has to determine a parameter *skipUpdate*(n) and a bit string *lowPassPartitioning*(n), where
    - the parameter *skipUpdate*(n) specifies whether the update steps shall be skipped for the current stage, and
    - the bit string *lowPassPartitioning*(n) specifies the partitioning of the given set of low-pass pictures $\{ L_n[0], \ldots, L_n[N_n-1] \}$ into the set of low-pass pictures $\{ L_{n+1}[0], \ldots, L_{n+1}[N_{n+1}-1] \}$ and the set of high-pass pictures $\{ H_{n+1}[0], \ldots, H_{n+1}[N_n-N_{n+1}-1] \}$.
  - The decomposition process specified in sec. 2.3.1 is invoked with the set of N_n low-pass pictures $\{ L_n[0], \ldots, L_n[N_n-1] \}$ and the parameters *skipUpdate*(n) and *lowPassPartitioning*(n) as input, and the output is assigned to the set of low-pass pictures $\{ L_{n+1}[0], \ldots, L_{n+1}[N_{n+1}-1] \}$ with reduced temporal resolution and the set of high-pass pictures $\{ H_{n+1}[0], \ldots, H_{n+1}[N_n-N_{n+1}-1] \}$. Furthermore, the decomposition process specified in sec. 2.3.1 determines a set of prediction data arrays $\{ M_{P,n+1}[0], \ldots, M_{P,n+1}[N_n-N_{n+1}-1] \}$ that are necessary to reconstruct the set of low-pass pictures $\{ L_n[0], \ldots, L_n[N_n-1] \}$.
  - The number of performed decomposition stages n is increased by 1: $n = n + 1$.

After decomposition, the given group of N_0 pictures $\{L_0\}$ is completely represented by N_n low-pass pictures $\{L_n\}$, $N_0 - N_n$ high-pass pictures $\{ \{H_1\}, \{H_2\}, \ldots, \{H_n\} \}$ and $N_0 - N_n$ prediction data arrays $\{ \{M_{P,1}\}, \{M_{P,2}\}, \ldots, \{M_{P,n}\} \}$ as well as the chosen parameters $\{$*skipUpdate*$\}$ and $\{$*lowPassPartitioning*$\}$.

At the decoder side, an approximation of the group of N_0 original pictures $\{ L'_0[0], \ldots, L'_0[N_0-1] \}$ can be reconstructed given the number of original pictures N_0, the number of decomposition stages n, approximations of the low-pass pictures $\{ L'_n[0], \ldots, L'_n[N_n-1] \}$ and the high-pass pictures $\{ H'_1[0], \ldots, H'_1[N_0-N_1-1] \}$, $\{ H'_2[0], \ldots, H'_2[N_1-N_2-1] \}$, …, $\{ H'_n[0], \ldots, H'_n[N_{n-1}-N_n-1] \}$, as well as the sets of prediction data arrays $\{ M_{P,1}[0], \ldots, M_{P,1}[N_0-N_1-1] \}$, $\{ M_{P,2}[0], \ldots, M_{P,2}[N_1-N_2-1] \}$, …,

{ $M_{P,n}[0]$, …, $M_{P,n}[N_{n-1}–N_n–1]$ } and control parameters {*skipUpdate*} and {*lowPassPartitioning*}. The reconstruction process is specified as follows.

- While the number of decomposition stages *n* is greater than 0, the following applies.
  - The reconstruction process specified in sec. 2.3.2 is invoked with the variable N_{n-1}, the parameters *skipUpdate*(*n*–1) and *lowPassPartitioning*(*n*–1), the ordered sets of reconstructed low-pass pictures { $L'_n[0]$, …, $L'_n[N_n–1]$ } and reconstructed high-pass pictures { $H'_n[0]$, …, $H'_n[N_{n-1}–N_n–1]$ }, and the prediction data arrays { $M_{P,n}[0]$, …, $M_{P,n}[N_{n-1}–N_n–1]$ } as input, and the output is assigned to the reconstructed of low-pass pictures { $L'_{n-1}[0]$, …, $L'_{n-1}[N_{n-1}–1]$ } with increased temporal resolution.

    Note that, given N_0 and the control parameters {lowPassPartitioning}, all variables N_k can be recursively determined. Given a variable N_k and the bit string lowPassPartitioning(k), the variable N_{k+1} is simply given by the number of ones in the bit string lowPassPartitioning(k).

  - The number of decomposition stages *n* is decreased by 1: $n = n – 1$.

2.3.1 Decomposition of a set of low-pass pictures

This section describes the decomposition of an ordered set of low-pass pictures into an ordered set of low-pass pictures with reduced temporal resolution and an ordered set of high-pass pictures.

Inputs to this process are an ordered set of N_k low-pass pictures { $L_k[0]$, …, $L_k[N_k – 1]$ }, a flag *skipUpdate*, and a bit string *lowPassPartitioning* of N_k bits specifying the partitioning of the given set of low-pass pictures into the set of low-pass pictures with reduced temporal resolution and the set of high-pass pictures.

Outputs of this process are an ordered set of N_{k+1} low-pass pictures { $L_{k+1}[0]$, …, $L_{k+1}[N_{k+1} – 1]$ }, an ordered set of $N_k – N_{k+1}$ high-pass pictures { $H_{k+1}[0]$, …, $H_{k+1}[N_k – N_{k+1} – 1]$ } as well as an ordered set of $N_k – N_{k+1}$ prediction data arrays { $M_{P,k+1}[0]$, …, $M_{P,k+1}[N_k – N_{k+1} – 1]$ } that are used in the prediction steps.

The construction process for reference index lists specified in sec. 2.3.3 is invoked with N_k and *lowPassPartitioning* as input, and the outputs are assigned to the lists of reference index lists { *refIdxList0*[0], …, *refIdxList0*[$N_k – 1$] } and { *refIdxList1*[0], …, *refIdxList1*[$N_k – 1$] } and the lists *mapHP2LP*[] and *mapLP2HP*[] that specify the mapping of picture indices.

The ordered set of high-pass pictures { $H_{k+1}[0]$, …, $H_{k+1}[N_k – N_{k+1} – 1]$ }, and the ordered set of prediction data arrays { $M_{P,k+1}[0]$, …, $M_{P,k+1}[N_k – N_{k+1} – 1]$ } are obtained as follows. Let *idxLP* and *idxHP* be two variables that are initially set to zero. While *idxLP* is less then N_k, the following applies:

- When *lowPassPartitioning*[*idxLP*] is equal to 0, the following applies.
  - A process for motion estimation and mode decision is invoked with the variable *idxLP*, the reference index lists *refIdxList0*[*idxLP*] and *refIdxList1*[*idxLP*], and the ordered set of low-pass pictures { $L_k[0]$, …, $L_k[N_k – 1]$ } as input, and the output is assigned to $M_{P,k+1}$[*idxHP*]. The actual algorithm that is used for determining the prediction data array $M_{P,k+1}$[*idxHP*] has to be optimised as part of the operational encoder control. An example method using Lagrangian optimisation techniques is described in sec. 4.2.
  - A prediction picture *P* is generated by invoking the process specified in sec. 2.3.4 with the variable *isUpdateFlag* = 0, the reference index lists *refIdxList0*[*idxLP*] and *refIdxList1*[*idxLP*], the prediction data array $M_{P,k+1}$[*idxHP*], and the ordered set of low-pass pictures { $L_k[0]$, …, $L_k[N_k –1]$ } as input.
  - The high-pass picture H_{k+1}[*idxHP*] is generated by

    $$h[i, j] = l[i, j] – p[i, j]$$

    where *h*[], *l*[], and *p*[] represent the luma and chroma sample arrays of the pictures H_{k+1}[*idxHP*], L_k[*idxLP*], and *P*, respectively.
  - The variable *idxHP* is incremented by 1: *idxHP* = *idxHP* + 1.

- The variable *idxLP* is incremented by 1: *idxLP* = *idxLP* + 1.

The ordered set of low-pass pictures with reduced temporal resolution { $L_{k+1}[0]$, …, $L_{k+1}[N_{k+1} - 1]$ } are obtained as follows. Let *idxLP* and *idxLPRR* be two variables that are initially set to zero. While *idxLP* is less then N_k, the following applies:

- When *lowPassPartitioning*[*idxLP*] is equal to 1, the following applies.
  - A prediction picture *P* is obtained as follows:
    - If the variable *skipUpdate* is equal to 0, the following applies.
      - The derivation process for prediction data arrays used in the update steps specified in sec. 2.3.5 is invoked with *idxLP*, the lists of reference index lists { *refIdxList0*[0], …, *refIdxList0*[$N_k - 1$] } and { *refIdxList1*[0], …, *refIdxList1*[$N_k - 1$] }, the ordered set of prediction data arrays (used in the prediction steps) { $M_{P,k+1}[0]$, …, $M_{P,k+1}[N_k - N_{k+1} - 1]$ }, and the picture index mapping list *mapHP2LP*[] as input, and the output is assigned to the prediction data array M_U.
      - The prediction picture *P* is generated by invoking the process specified in sec. 2.3.4 with the variable *isUpdateFlag* = 1, the reference index lists *refIdxList0*[*idxLP*] and *refIdxList1*[*idxLP*], the prediction data array M_U, the ordered set of high-pass pictures { $H_{k+1}[0]$, …, $H_{k+1}[N_k - N_{k+1} - 1]$ }, and the ordered set of prediction data arrays (used in the prediction steps) { $M_{P,k+1}[0]$, …, $M_{P,k+1}[N_k - N_{k+1} - 1]$ } as input.
    - Otherwise (*skipUpdate* is equal to 1), all samples of the prediction picture *P* are set to zero.
  - The low-pass picture L_{k+1}[*idxLPRR*] is generated by

    $$l_l[i, j] = l[i, j] + (p[i, j] >> 1)$$

    where l_l[], *l*[], and *p*[] represent the luma and chroma sample arrays of the pictures L_{k+1}[*idxLPRR*], L_k[*idxLP*], and *P*, respectively.
  - The variable *idxLPRR* is incremented by 1: *idxLPRR* = *idxLPRR* + 1.
- The variable *idxLP* is incremented by 1: *idxLP* = *idxLP* + 1.

2.3.2 Reconstruction of a set of low-pass pictures

This section describes the reconstruction of an ordered set of low-pass pictures from an ordered set of low-pass pictures with reduced temporal and an ordered set of high-pass pictures.

Inputs to this process are a variable N_k, a flag *skipUpdate*, a bit string *lowPassPartitioning* of N_k bits, an ordered set of N_{k+1} low-pass pictures { $L_{k+1}[0]$, …, $L_{k+1}[N_{k+1} - 1]$ }, an ordered set of $N_k - N_{k+1}$ high-pass pictures { $H_{k+1}[0]$, …, $H_{k+1}[N_k - N_{k+1} - 1]$ } as well as an ordered set of $N_k - N_{k+1}$ prediction data arrays { $M_{P,k+1}[0]$, …, $M_{P,k+1}[N_k - N_{k+1} - 1]$ } that are used in the prediction steps.

Output of this process is an ordered set of N_k low-pass pictures { $L_k[0]$, …, $L_k[N_k - 1]$ }.

The construction process for reference index lists specified in sec. 2.3.3 is invoked with N_k and *lowPassPartitioning* as input, and the outputs are assigned to the lists of reference index lists { *refIdxList0*[0], …, *refIdxList0*[$N_k - 1$] } and { *refIdxList1*[0], …, *refIdxList1*[$N_k - 1$] } and the lists *mapHP2LP*[] and *mapLP2HP*[] that specify the mapping of picture indices.

In the first stage, a subset of the low-pass pictures { $L_k[0]$, …, $L_k[N_k - 1]$ } is reconstructed as follows. Let *idxLP* and *idxLPRR* be two variables that are initially set to zero. While *idxLP* is less then N_k, the following applies:

- When *lowPassPartitioning*[*idxLP*] is equal to 1, the following applies.
  - A prediction picture *P* is obtained as follows:
    - If the variable *skipUpdate* is equal to 0, the following applies.
      - The derivation process for prediction data arrays used in the update steps specified in sec. 2.3.5 is invoked with *idxLP*, the lists of reference index lists { *refIdxList0*[0], …,

refIdxList0[$N_k - 1$] } and { *refIdxList1*[0], …, *refIdxList1*[$N_k - 1$] }, the ordered set of prediction data arrays (used in the prediction steps) { $M_{P,k+1}[0]$, …, $M_{P,k+1}[N_k - N_{k+1} - 1]$ }, and the picture index mapping list *mapHP2LP*[] as input, and the output is assigned to the prediction data array M_U.

- The prediction picture P is generated by invoking the process specified in sec. 2.3.4 with the variable *isUpdateFlag* = 1, the reference index lists *refIdxList0*[*idxLP*] and *refIdxList1*[*idxLP*], the prediction data array M_U, the ordered set of high-pass pictures { $H_{k+1}[0]$, …, $H_{k+1}[N_k - N_{k+1} - 1]$ }, and the ordered set of prediction data arrays (used in the prediction steps) { $M_{P,k+1}[0]$, …, $M_{P,k+1}[N_k - N_{k+1} - 1]$ } as input.

▪ Otherwise (*skipUpdate* is equal to 1), all samples of the prediction picture P are set to zero.

o The low-pass picture L_k[*idxLP*] is generated by

$$l[i, j] = l_l[i, j] - (p[i, j] >> 1)$$

where l [], l_l[], and p[] represent the luma and chroma sample arrays of the pictures L_k[*idxLP*], L_{k+1}[*idxLPRR*], and P, respectively.

o The variable *idxLPRR* is incremented by 1: *idxLPRR* = *idxLPRR* + 1.

- The variable *idxLP* is incremented by 1: *idxLP* = *idxLP* + 1.

The remaining subset of the low-pass pictures { $L_k[0]$, …, $L_k[N_k - 1]$ } is reconstructed as follows. Let *idxLP* and *idxHP* be two variables that are initially set to zero. While *idxLP* is less then N_k, the following applies:

- When *lowPassPartitioning*[*idxLP*] is equal to 0, the following applies.
  - o A prediction picture P is generated by invoking the process specified in sec. 2.3.4 with the variable *isUpdateFlag* = 0, the reference index lists *refIdxList0*[*idxLP*] and *refIdxList1*[*idxLP*], the prediction data array $M_{P,k+1}$[*idxHP*], and the ordered set of low-pass pictures { $L_k[0]$, …, $L_k[N_k - 1]$ } as input.
  - o The low-pass picture L_k[*idxLP*] is generated by

$$l[i, j] = h[i, j] + p[i, j]$$

where l[], h[], and p[] represent the luma and chroma sample arrays of the pictures L_k[*idxLP*], H_{k+1}[*idxHP*], and P, respectively.

  - o Subsequently, the deblocking filter process specified in sec. 2.3.6 is invoked with L_k[*idxLP*], $M_{P,k+1}$[*idxHP*], and the array $C_{H,k+1}$[*idxHP*], which specifies the quantisation parameters and transform coefficient levels of the highest SNR layer for the high-pass picture H_{k+1}[*idxHP*], as input, and the output is assigned to L_k[*idxLP*].
  - o The variable *idxHP* is incremented by 1: *idxHP* = *idxHP* + 1.
- The variable *idxLP* is incremented by 1: *idxLP* = *idxLP* + 1.

2.3.3 Construction of reference index lists

This section describes the process for the derivation of the reference index lists.

Inputs to this process are the number of original or reconstructed low-pass pictures N_k and the bit string *lowPassPartitioning* of N_k bits.

Outputs of this process are two lists of reference index lists { *refIdxList0*[0], …, *refIdxList0*[$N_k - 1$] } and { *refIdxList1*[0], …, *refIdxList1*[$N_k - 1$] } as well as two lists *mapHP2LP*[] and *mapLP2HP*[] that specify the mapping of picture indices.

If *lowPassPartitioning*[i] is equal to 0, the reference index lists *refIdxList0*[i] and *refIdxList1*[i] specify the mapping of the list 0 and list 1 reference indices, respectively, onto the entries of the ordered list of original or reconstructed low-pass pictures { $L_k[0]$, …, $L_k[N_k - 1]$ } that are used for motion-compensated prediction in the prediction steps. Otherwise (if *lowPassPartitioning*[i] is equal to 1), the reference index

lists *refIdxList0*[*i*] and *refIdxList1*[*i*] specify the mapping of the list 0 and list 1 reference indices, respectively, onto the entries of the ordered list of high-pass pictures { $H_{k+1}[0], \ldots, H_{k+1}[N_k - N_{k+1} - 1]$ } that are used for motion-compensated update in the update steps.

The lists *mapHP2LP*[] and *mapLP2HP*[] are derived by the following procedure.

```
idxHP = 0
for( j = 0; j < Nk; j++ )
  if( lowPassPartitioning[j] == 0 ) {
    mapLP2HP[j]      = idxHP
    mapHP2LP[idxHP] = j
    idxHP            = idxHP + 1
  }
```

The initial reference index lists *refIdxList0*[*i*] and *refIdxList1*[*i*] with *i* varying from 0 to $N_k - 1$, inclusive, are generated as follows.

- If *lowPassPartitioning*[*i*] is equal to 0, the following applies
  - The initial reference index list *refIdxList0*[*i*] is constructed by the following procedure.

```
idx = 0
for( j = i + 1; j < Nk; j++ )
  if( lowPassPartitioning[j] == 1 )
    refIdxList0[i][idx++] = j
for( j = 0; j < i; j++ )
  if( lowPassPartitioning[j] == 1 )
    refIdxList0[i][idx++] = j
```

  - The initial reference index list *refIdxList1*[*i*] is constructed by the following procedure.

```
idx = 0
for( j = i - 1; j >= 0; j-- )
  if( lowPassPartitioning[j] == 1 )
    refIdxList1[i][idx++] = j
for( j = Nk - 1; j > i; j-- )
  if( lowPassPartitioning[j] == 1 )
    refIdxList0[i][idx++] = j
```

- Otherwise (*lowPassPartitioning*[*i*] is equal to 1), the following applies
  - The initial reference index list *refIdxList0*[*i*] is constructed by the following procedure.

```
idx = 0
for( j = i + 1; j < Nk; j++ )
  if( lowPassPartitioning[j] == 0 )
    refIdxList0[i][idx++] = mapLP2HP[j]
for( j = 0; j < i; j++ )
  if( lowPassPartitioning[j] == 0 )
    refIdxList0[i][idx++] = mapLP2HP[j]
```

  - The initial reference index list *refIdxList1*[*i*] is constructed by the following procedure.

```
idx = 0
for( j = i - 1; j >= 0; j-- )
  if( lowPassPartitioning[j] == 0 )
    refIdxList1[i][idx++] = mapLP2HP[j]
for( j = Nk - 1; j > i; j-- )
  if( lowPassPartitioning[j] == 0 )
    refIdxList1[i][idx++] = mapLP2HP[j]
```

Initially, the reference index lists *refIdxList0*[*i*] and *refIdxList1*[*i*] that are used in the prediction steps (*lowPassPartitioning*[*i*] is equal to 0) have N_{k+1} entries. For these reference lists, a reordering process similar to that specified in H.264/AVC [1] can be applied. The number of active entries is determined by syntax elements that are set by the encoder and are transmitted in the slice header.

Since, the prediction data arrays M_u that are used in the update steps are not transmitted, no reordering is designated for the reference index lists *refIdxList0*[*i*] and *refIdxList1*[*i*] that are used in the update steps (*lowPassPartitioning*[*i*] is equal to 1). For these reference lists *refIdxList0*[*i*] and *refIdxList1*[*i*] the number of active entries is set to the number of initial entries, which is equal to $N_k - N_{k+1}$.

2.3.4 General prediction process

This section describes a general prediction process, which is used by the prediction and update steps at both the analysis and synthesis side.

Inputs to this process are a variable *isUpdateFlag* specifying if the output picture P is used in a prediction or update step, the reference index lists *refIdxList0* and *refIdxList1*, and the prediction data array M as well as a list of low-pass pictures $\{ L_k[0], \ldots, L_k[N_k - 1] \}$ in case the variable *isUpdateFlag* is equal to 0 or a list of high-pass pictures $\{ H_{k+1}[0], \ldots, H_{k+1}[N_k - N_{k+1} - 1] \}$ and a list of prediction data arrays $\{ M_{u,k+1}[0], \ldots, M_{u,k+1}[N_k - N_{k+1} - 1] \}$ in case the variable *isUpdateFlag* is equal to 1.

Output of this process is a motion-compensated prediction signal or prediction picture P.

Let Y_P, U_P, and V_P be the arrays of luma (Y) and chroma (U, V) samples of the prediction picture P.

If the flag *isUpdateFlag* is equal to 0, let $Y_R[l]$, $U_R[l]$, and $V_R[l]$ be the arrays of luma (Y) and chroma (U, V) samples of the low-pass picture $L_k[l]$; otherwise (the flag *isUpdateFlag* is equal to 1), let $Y_R[l]$, $U_R[l]$, and $V_R[l]$ be the arrays of luma (Y) and chroma (U, V) samples of the high-pass picture $H_{k+1}[l]$.

The prediction signal P is constructed in a macroblock-wise manner as described in the following:

- If the macroblock mode specified in the prediction data array M is equal to INTRA, all luma and chroma samples $Y_P[i,j]$, $U_P[i,j]$, and $V_P[i,j]$ of the prediction picture P that are covered by the regarded macroblock are set to zero:

$$Y_P[i, j] = 0$$
$$U_P[i, j] = 0$$
$$V_P[i, j] = 0$$

- Otherwise (the macroblock mode specified in the prediction data array M is not equal to INTRA), for each macroblock or sub-macroblock partition the following applies:
  - A variable *predFlagL0* is set equal to 1 if the prediction data M specifies list 0 prediction or bi-prediction for the current macroblock partition; otherwise *predFlagL0* is set equal to 0. Similarly, a variable *predFlagL1* is set equal to 1 if the prediction data array M specifies list 1 prediction or bi-prediction for the current macroblock partition; otherwise, predFlagL1 is set equal to 0.
  - If *predFlagLN* (with N being replaced by 0 or 1) is equal to 1, a list N prediction signal for the luma and chroma samples of the current macroblock or sub-macroblock partition is obtained by motion-compensated prediction as specified in [1]:

$$Y_{\text{PredLN}}[i, j] = \text{Int}_Y\left(Y_R[refIdxListN[r_N]], i + m_{Nx}, j + m_{Ny}\right)$$
$$U_{\text{PredLN}}[i, j] = \text{Int}_C\left(U_R[refIdxListN[r_N]], i + m_{Nx}, j + m_{Ny}\right)$$
$$V_{\text{PredLN}}[i, j] = \text{Int}_C\left(V_R[refIdxListN[r_N]], i + m_{Nx}, j + m_{Ny}\right)$$

    $\mathbf{m}_0 = [m_{0x}, m_{0y}]^T$ and $\mathbf{m}_1 = [m_{1x}, m_{1y}]^T$ are the list 0 and list 1 motion vectors of the current macroblock or sub-macroblock partition, respectively; and r_0 and r_1 are the list 0 and list 1 reference indices of the current macroblock partition, respectively. $\mathbf{m}_0$, $\mathbf{m}_1$, r_0, and r_1 are specified by the prediction data array M.

    The function $\text{Int}_Y(Y[l], i, j)$ represents the interpolation process for motion-compensated prediction of luma samples given the luma sample array $Y[l]$ and the quarter-sample accurate luma location (i, j). Similarly, the function $\text{Int}_C(C[l], i, j)$ represents the interpolation process for motion-compensated prediction of chroma samples given the chroma sample array $C[l]$ and the eighth-sample accurate chroma location (i, j). In case *isUpdateFlag* is equal to 0, both the luma and chroma interpolation process are identical to the corresponding interpolation processes specified in H.264/AVC [1]. In case *isUpdateFlag* is equal to 1, the interpolation processes differ from the interpolation processes specified in [1] in the following two points:

    - The clipping to the interval [0; 255] is removed.

- Sample values inside the sample arrays $Y[l]$ and $C[l]$ that belong to a macroblock for which the INTRA macroblock mode is specified in the associated prediction data array $M_{P,k+1}[l]$ are set to zero for the usage in the interpolation process. Note, that these samples are also set to zero when the luma or chroma location (i, j) represents a sample-accurate location.

o Given the flags *predFlagL0*, *predFlagL1*, the list 0 prediction signals Y_{predL0}, U_{predL0}, V_{predL0} and/or the list 1 prediction signals Y_{predL1}, U_{predL1}, V_{predL1}, the luma and chroma samples for the current macroblock or sub-macroblock partition are obtained by the weighted sample prediction process as specified in H.264/AVC [1] with the only difference that the clipping to the interval [0; 255] is removed in case the variable *isUpdateFlag* is equal to 1. In general, the prediction weights can be adjusted to reflect intensity changes in a video sequence. For the default weighted sample prediction process, the luma and chroma samples $C_P[i, j]$ (with C representing Y, U, or V) of the prediction picture P that are covered by the regarded macroblock or sub-macroblock partition are derived as follows:

- If *predFlagL0* is equal to 1 and *predFlagL1* is equal to 1,

$$C_P[i, j] = \left(C_{\text{PredL0}}[i, j] + C_{\text{PredL1}}[i, j] + 1\right) >> 1$$

- Otherwise, if *predFlagL0* is equal to 1,

$$C_P[i, j] = C_{\text{PredL0}}[i, j]$$

- Otherwise (*predFlagL1* is equal to 1),

$$C_P[i, j] = C_{\text{PredL1}}[i, j]$$

As it can be seen from the above description, the general process for generating (motion-compensated) prediction pictures P is nearly identical to the reconstruction process of B slices (prior to the deblocking filter) as specified in H.264/AVC [1]. The following differences can be identified:

- If the variable *isUpdateFlag* is equal to 1, the clipping to the interval [0; 255] is removed in the processes of sample interpolation and weighted sample prediction.
- Simplified INTRA mode reconstruction with all samples set to zero.
- Simplified reconstruction for motion-compensated prediction modes without residual information.
- Simplified derivation process for the motion vectors and reference indices used in the direct macroblock and sub-macroblock mode (see sec. 2.2).

2.3.5 Derivation of prediction data for the update steps

This section describes the derivation of prediction data arrays M_U for the usage in the update steps from the set of prediction data arrays $\{M_P\}$ used in the prediction steps.

Inputs to the process are *idxLP*, the lists of reference index lists { *refIdxList0*[0], …, *refIdxList0*[$N_k - 1$] } and { *refIdxList1*[0], …, *refIdxList1*[$N_k - 1$] }, and the ordered set of prediction data arrays { $M_{P,k+1}[0]$, …, $M_{P,k+1}[N_k - N_{k+1} - 1]$ } that are used in the prediction steps.

Output of this process is a prediction data array M_U that is used in an update step.

Let *mvUpdateL0*[][], *mvUpdateL1*[][] and *numConnectedL0*[][], *numConnectedL1*[][] be four arrays that represent the motion vector candidates and the associated number of connected luma samples, respectively, for the prediction data array M_u. Each entry *mvUpdateL0*$[i, j][r]$ (*mvUpdateL1*$[i, j][r]$) specifies a list 0 (list 1) motion vector candidate for the 4x4 luma block with the block coordinate (i, j) and the list 0 (list 1) reference index r, the associates number of connected luma samples is represented by the entry *numConnectedL0*$[i, j][r]$ (*numConnectedL1*$[i, j][r]$). The arrays *mvUpdateLX*[][] and *numConnectedLX*[][] (with X being replace by 0 or 1) are determined by the following algorithm.

- Initially, all entries of the array *numConnectedLX*[][] are set to zero, and a reference index r_X is set to zero.
- While the reference index r_X is less than $N_k - N_{k+1}$, the following applies.

- Let Y be a template variable that is derived by $Y = 1 - X$.
- Let *idxHP* be a variable that is equal to *refIdxListX*[*idxLP*][r_X].
- Let *refIdxListCurr*[] be the reference index list specified by *refIdxListY*[*mapHP2LP*[*idxHP*]].
- The macroblocks of the prediction data array $M_{P,k+1}$[*idxHP*] are processed in raster-scan order, the 4x4 luma blocks of each macroblock are processed in scan order, and for each 4x4 luma block, the following applies.
  - When the current 4x4 luma block belongs to a motion-compensated partition that specifies list Y prediction or bi-prediction, let r_Y be the associated list Y reference index; otherwise, r_Y is set equal to –1.
  - If r_Y is not equal to –1 and *refIdxListCurr*[r_Y] is equal to *idxLP*, the following applies.
    - Let (i_P, j_P) be the block coordinate (in units of 4 luma samples) of the regarded 4x4 luma block of the prediction data array $M_{P,k+1}$[*idxHP*], and let $\mathbf{m}_P = [\ m_{Px}, m_{Py}\ ]^T$ be the associated list Y motion vector.
    - A luma location (i_Y, j_Y) is derived by

      $i_{lum} = (i_P << 2) + ((m_{Px} + 2) >> 2)$
      $j_{lum} = (j_P << 2) + ((m_{Py} + 2) >> 2)$

    - Let (di, dj) be a luma location difference. For each (di, dj) of the ordered set { (0, 0), (4, 0), (0, 4), (4, 4) }, for which the luma location $(i_{lum}+ di, j_{lum}+ dj)$ specifies a luma sample inside the picture area, the following applies.
      - Let (i_U, j_U) be a block coordinate (in units of 4 luma samples) that is derived by

        $i_U = (i_{lum} + di) >> 2$
        $j_U = (j_{lum} + dj) >> 2$

      - Let *numSamples* be a variable that is calculated as follows

        $numSamples = (di > 0 \ ? \ i_{lum} \ \% \ 4 : 4 - (i_{lum} \ \% \ 4)) \times (dj > 0 \ ? \ j_{lum} \ \% \ 4 : 4 - (j_{lum} \ \% \ 4))$

      - If $mvUpdateLX[i_U, j_U][r_X]$ is equal to $-\mathbf{m}_P$, the following applies

        $numConnectedLX[i_U, j_U][r_X]$ += *numSamples*

      - Otherwise, if *numSamples* is greater than $numConnectedLX[i_U, j_U][r_X]$, the following applies.

        $mvUpdateLX\ [i_U, j_U][r_X] = -\mathbf{m}_P$
        $numConnectedLX[i_U, j_U][r_X]$ += *numSamples*

- The reference index r_X is incremented by 1: $r_X = r_X + 1$.

After deriving the arrays *mvUpdateL0*[][], *mvUpdateL1*[][], *numConnectedL0*[][], *numConnectedL1*[][], the macroblock modes, sub-macroblock modes, reference indices, and motion vectors of the prediction data array M_U are determined. Therefore, for each macroblocks of the prediction data arrays M_U, the following applies.

- Let *intraFlag* be a Boolean variable that is initially set to 0.
- For each sub-macroblock (8x8 luma block) of the current macroblock, the following applies.
  - Let (i, j) be the block coordinate of the upper-left 4x4 luma of the current sub-macroblock.
  - Let r_X (with X being replaced with 0 or 1) be the list X reference index for the current sub-macroblock. r_X is determined by the following procedure.

```
rx = -1
numMaxSamples = 16
for( r = 0; r < Nk – Nk+1; r++ ) {
  numSamples = numConnectedLX[i  ,j  ][r] +
               numConnectedLX[i  ,j+1][r] +
               numConnectedLX[i+1,j  ][r] +
```

```
                numConnectedLX[i+1,j+1][r]
    if( numSamples > numMaxSamples ) {
      r_x = r
      numMaxSamples = numSamples
    }
  }
```

- The sub-macroblock type and the corresponding reference indices of the current sub-macroblock are determined as follows.
  - If r_0 is greater than –1 and r_1 is greater than –1, the sub-macroblock type is set to B_Bi_4x4, and the corresponding list 0 and list 1 reference indices are set to r_0 and r_1, respectively.
  - Otherwise, if r_0 is greater than –1 and r_1 is equal to –1, the sub-macroblock type is set to B_L0_4x4, and the corresponding list 0 reference index is set to r_0.
  - Otherwise, if r_0 is equal to –1 and r_1 is greater than –1, the sub-macroblock type is set to B_L1_4x4, and the corresponding list 1 reference index is set to r_1.
  - Otherwise (r_0 is equal to –1 and r_1 is equal to –1), the sub-macroblock type is marked as undefined and the variable *intraFlag* is set to 1.
- If r_X (with X being replaced with 0 or 1) is not equal to –1, the list X motion vector $\mathbf{m}(n)$ of each 4x4 luma block with the block index $n \in \{ 0, 1, 2, 3 \}$ of the current sub-macroblock is determined as follows.
  - If *numConnectedLX*[$i + (n \% 2), j + (n / 2)$][r_X] is greater than zero, the motion vector $\mathbf{m}(n)$ is set equal to *mvUpdateLX*[$i + (n \% 2), j + (n / 2)$][r_X].
  - Otherwise (*numConnectedLX*[$i + (n \% 2), j + (n / 2)$][r_X] is equal to zero), the motion vector $\mathbf{m}(n)$ is set equal to a motion vector $\mathbf{m}_{pred}$, which is derived by the following procedure.

```
maxConnected = 0
for( jj = j; jj < j + 2; jj++ )
for( ii = i; ii < i + 2; ii++ )
  if( numConnectedLX[ii,jj][r_x] > maxConnected )
    m_pred = mvUpdateLX[ii,jj][r_x]
```

- If *intraFlag* is equal to 0, the macroblock mode is set equal to B_8x8, otherwise, the macroblock mode is set equal to INTRA.

In many cases, it is possible to further summarize the 4x4 sub-macroblock and 8x8 macroblock partitions in order to obtain larger partitions for the motion compensation process. However, since the prediction data arrays M_U that are used in the update steps are not transmitted, a further summarisation will lead to identical encoding / decoding results.

2.3.6 Deblocking filter process

This section specifies the deblocking filter process.

Inputs to this process are a low-pass picture L_k, a prediction data array $M_{P,k+1}$, and an array $C_{H,k+1}$ specifying quantisation parameters and transform coefficient levels for each macroblock of the low-pass picture L_k.

Output of the process is a modified low-pass picture L_k.

The deblocking filter process for the picture L_k is applied as specified in the H.264/AVC standard [1], where

- the macroblock modes, the sub-macroblock modes, the reference indices, and the motion vectors are extracted from the given prediction data array $M_{P,k+1}$, and
- the transform coefficient levels and quantisation parameters are extracted from the array $C_{H,k+1}$.

3 Scalable Coding Scheme as Extension of H.264

Using the temporal decomposition scheme presented in sec. 2 with n decomposition stages, a group of N_0 input pictures is decomposed into $N_n > 0$ low-pass pictures and $N_0 - N_n$ high-pass pictures. The reconstruction process for the group of input pictures is specified by $N_0 - N_n$ prediction data arrays M_P that are used in the prediction steps (and for the derivation of the prediction data arrays M_U used in the update steps) as well as several control parameters as the n bit strings {*lowPassPartitioning*}, the n flags {*skipUpdate*}, and the number of active reference indices for each prediction data array M_P (see sec. 2). Thus, beside the control parameters (that are coded as part of the slice headers), the $N_0 - N_n$ prediction data arrays M_P, and approximations of the N_n low-pass pictures and the $N_0 - N_n$ high-pass pictures need to be transmitted. To map these data to NAL units, we use subsets of the slice layer syntax of H.264/AVC.

At first, in sec. 3.1, a single layer coding scheme and its relation to standard H.264/AVC coding is described. The extension to a scalable coding scheme is presented in sec. 3.2.

3.1 Single Layer Coding

3.1.1 Coding of prediction data arrays

As it is shown in sec. 2, the specification of the prediction data arrays M_P is nearly identical to the specification of the prediction information that is used in B slices of H.264/AVC. Consequently, the prediction arrays M_P are generally coded using a subset of the B slice syntax of H.264/AVC [1], which we named **M slice** syntax. For each macroblock, the following syntax elements are transmitted: the macroblock mode `mb_type`, the sub-macroblock modes `sub_mb_type` (if applicable), the reference indices `ref_idx_l0` and/or `ref_idx_l1` (if applicable), the motion vector differences `mvd_l0` and/or `mvd_l1` (if applicable), and the flag `end_of_slice_flag`. The motion vector predictors are derived as specified in the standard. The following differences to the coding of B slices in H.264/AVC can be identified:

- The syntax of the slice header is modified.
- The syntax element indicating if a macroblock is coded in skip mode (`mb_skip_run` or `mb_skip_flag`) is not transmitted.
- Only one intra mode (INTRA) is included in the set of possible macroblock modes. For signalling this intra mode, the codeword / binarization of the INTRA_4x4 mode is used. For this intra mode, no intra prediction modes are transmitted.
- No residual information (including the syntax elements `coded_block_pattern` and `mb_qp_delta`) is transmitted.
- The derivation process for the reference indices and motion vectors for the direct macroblock and sub-macroblock mode is modified (see sec. 2).

As mentioned in sec. 2, the encoder can select the number of active entries for the reference index list *refIdxList0* and *refIdxList1*. In our coding scheme it is also allowed that the number of active entries is set to zero for one (but not both) reference index lists. In that case, which is signalled in the slice header, the syntax elements `mb_type` and `sub_mb_type` are transmitted using the codewords (or the binarization and the context variables) that are specified for P slice coding in H.264/AVC.

3.1.2 Coding of high-pass pictures

In general, a high-pass picture H contains intra and residual macroblocks, where the location of the intra macroblocks is specified by the corresponding prediction data array M_P. Since the residual macroblocks represent prediction errors, the residual coding as specified in the H.264/AVC standard including transformation, scaling, and quantisation is employed. The intra macroblocks represent original samples, and thus the intra coding as specified in H.264/AVC including intra prediction and transformation, scaling, and quantisation of the residual signal is employed. For the coding of intra macroblocks, all intra macroblock modes (INTRA_4x4, INTRA_16x16, I_PCM) defined in H.264/AVC can be used. However,

since intra macroblocks should not be predicted from neighbouring residual macroblocks, the intra prediction is always performed as if the flag `contrained_intra_pred_flag` defined in the picture parameter set is equal to 1.

The high-pass pictures are coded using a subset of the slice syntax of H.264/AVC, which is named **H slice** syntax. For each macroblock, the following syntax elements are transmitted:

- If the macroblock is intra macroblock (as specified by the corresponding prediction data array M_P),
  - the macroblock mode `mb_type`,
  - the intra prediction modes,
- the residual information including the syntax elements `coded_block_pattern` and `mb_qp_delta` (if applicable), and
- the flag `end_of_slice_flag`

The macroblock mode `mb_type` is transmitted using the codewords (or the binarization and the context variables) that are specified for I slice coding in H.264/AVC.

The reconstruction process for H slices is nearly identical to the reconstruction process of P and B slices as it is specified in the H.264/AVC standard with the main difference that the (motion-compensated) prediction signal for residual blocks is always set to zero.

Since the information about the location of intra and residual macroblocks is not transmitted in an H slice, the H slices can only be decoded if the corresponding prediction data array M_P was already received. However, we believe that this is no disadvantage, since the high-pass pictures cannot be used in the reconstruction process without the corresponding prediction data arrays. Actually, by analysing the presented M and H slice syntax descriptions it can be seen that they can be combined into an MH slice syntax, which is nearly identical to the B slice syntax specified in the standard (the main difference is that the syntax element that signals the skip mode is not transmitted).

3.1.3 Coding of low-pass pictures

Low-pass pictures can be interpreted as original pictures, and thus they are generally coded using the syntax of H.264/AVC as specified in the standard. In the simplest version, all low-pass pictures are coded independently as intra pictures (using I slices only). In a more general version, only the first low-pass picture of a group of pictures (GOP) is coded as intra (IDR) picture and all remaining low-pass pictures inside GOP are coded as predictive pictures using any combination of I, P, and B slices, where preceding low-pass pictures of the same GOP can be used as reference for motion-compensated prediction. In case, the decomposition of a GOP is performed in a way that more than one low-pass picture are obtained, this scheme provides an improved coding efficiency while still allowing random access at the GOP level.

Especially for sequences with high spatial detail and slow motion, the coding efficiency can be improved further if the correlations between successive GOP's are exploited. Thus, in general, all low-pass pictures are coded as predictive pictures (I, P, or B slices) using reconstructed low-pass pictures including those of previous GOP's as reference; IDR (intra) pictures are inserted in regular intervals only to provide random access points. At the decoder side, low-pass pictures are parsed and reconstructed as specified in the H.264/AVC standard including the deblocking filter operation.

It is worth noting that if the number of decomposition stages n for all groups of pictures is set to 0 (which is always the case if the video is processed in groups of only 1 picture), the bit stream syntax of the presented coding scheme is identical to the bit stream syntax of H.264/AVC (possibly with the exception of some header information). Thus, the standard H.264/AVC coding can be interpreted as a special case of the presented coding scheme.

3.2 Scalable Coding

For a better understanding, the temporal, SNR (quality), and spatial scalability are described as separate features in the sec. 3.2.1, 3.2.2, and 3.2.3. The general concept of the scalable extension providing combined scalability is presented in sec. 3.2.4.

3.2.1 Temporal Scalability

As already mentioned in sec. 2.2, the temporal decomposition framework presented in sec. 2 inherently provides temporal scalability. By using n decomposition stages, up to n levels of temporal scalability can be provided.

In Figure 5, an example for the temporal decomposition of a group of 12 pictures using 3 decomposition stages is illustrated (the same example was depicted in Figure 4). If only the low-pass pictures $\{L\}^3$ that are obtained after the third (highest) decomposition stage are transmitted, the picture sequence $\{L\}^{3*}$ that can be reconstructed at the decoder side has 1/12 of the temporal resolution of the input sequence. The picture sequence $\{L\}^3$ is also referred to as temporal base layer. By additionally transmitting the high-pass pictures $\{H\}^3$ and the corresponding prediction data arrays $\{M_P\}^3$, the decoder can reconstruct an approximation of the picture sequence $\{L\}^2$ that has 1/4 of the temporal resolution of the input sequence. The high-pass pictures $\{H\}^3$ and the corresponding prediction data arrays $\{M_P\}^3$ are also referred to as the first temporal enhancement layer. By further adding the high-pass pictures $\{H\}^2$ and the prediction data arrays $\{M_P\}^2$, a picture sequence $\{L\}^{1*}$ with half the temporal resolution can be reconstructed. And finally, if the also the remaining high-pass pictures and prediction data arrays $\{H\}^1$ and $\{M_P\}^1$ are transmitted, a reconstructed version of the original input sequence with the full temporal resolution is obtained.

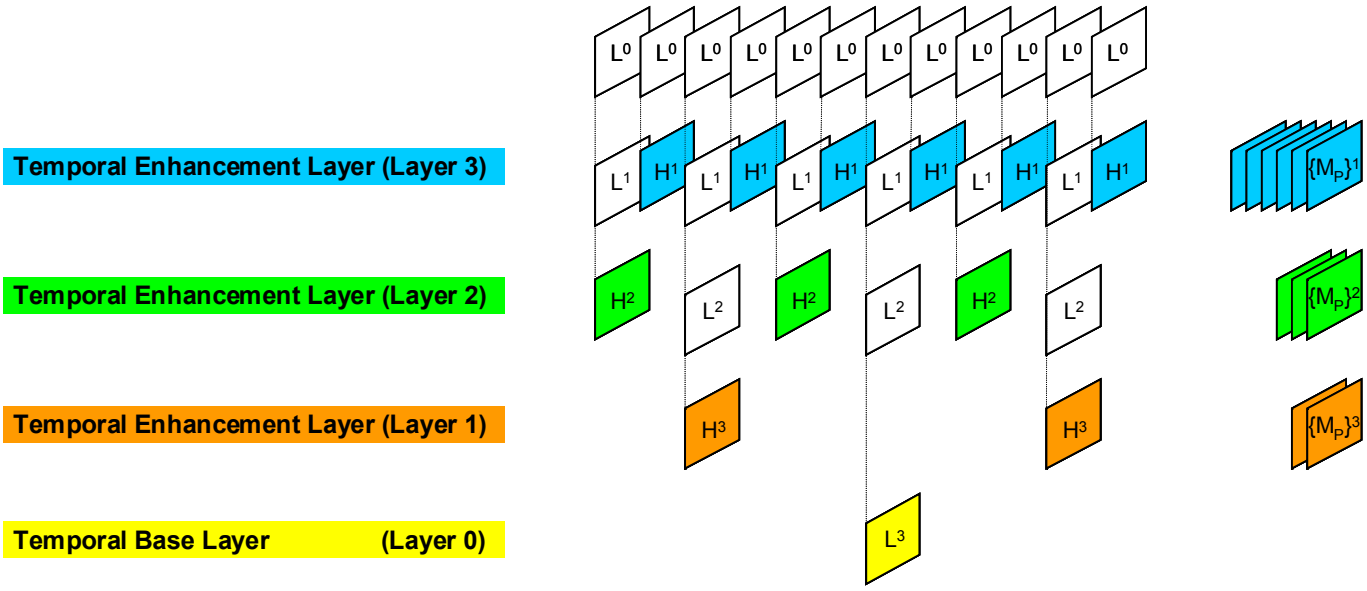

Figure 5: Illustration of temporal scalability.

In general, by using n decomposition stages, the decomposition structure can be designed in a way that n levels of temporal scalability are provided with temporal resolution conversion factors of $1/m_0$, $1/(m_0 \cdot m_1)$, …, $1/(m_0 \cdot m_1 \cdot \ldots \cdot m_{n-1})$, where m_i represents any integral number greater than 1. Therefore, a picture sequence has to be coded in groups of $N_0 = (j \cdot m_0 \cdot m_1 \cdot \ldots \cdot m_{n-1})$ pictures with j being an integral number greater than 0 (the GOP size does not need to be constant within the picture sequence).

If the sampling period between successive pictures of a reconstructed picture sequences does not need to be constant, a temporal scalability with (average) temporal resolution conversion factors of k_0/m_0, $(k_0 \cdot k_1)/(m_0 \cdot m_1)$, …, $(k_0 \cdot k_1 \cdot \ldots \cdot k_{n-1})/(m_0 \cdot m_1 \cdot \ldots \cdot m_{n-1})$ is possible (cp. sec. 2.2), where k_i represents any integral number greater than 0 and m_i represents any integral number greater than k_i and less than $2 \cdot k_i$. As in the case described above, a picture sequence has to be coded in groups of $N_0 = (j \cdot m_0 \cdot m_1 \cdot \ldots \cdot m_{n-1})$ pictures with j being an integral number greater than 0.

It should be mentioned that a similar degree of temporal scalability could be realized with standard H.264/AVC coding (or the presented coding scheme with GOP's of only 1 picture) by using sub-sequences and/or regularly inserted non-reference pictures.

3.2.2 SNR (Quality) Scalability

The open-loop structure of the presented subband approach (see sec. 2) provides the possibility to efficiently incorporate SNR scalability. For obtaining an SNR scalable representation of a group of pictures, the concept that has already been used in previous video coding standards as H.262/MPEG-2 Visual [7], H.263 [8], or MPEG-4 Visual [9] is adapted to the subband structure.

In Figure 6, the general concept of our SNR scalable coding scheme is depicted. A group of pictures is decomposed into low- and high-pass pictures as described in sec. 2. A base layer representation of the subband pictures, and thus of the group of input pictures, is obtained by coding the low- and high-pass pictures L and H as well as the corresponding prediction data arrays M_P as described in sec. 3.1. In each enhancement layer, approximations of the residual signals computed between the original subband pictures obtained by the analysis filterbank and the reconstructed subband pictures obtained after decoding the base layer (and previous enhancement layers) are transmitted.

For encoding the enhancement representations, we define an additional macroblock mode called INTRA_BASE. The reconstruction process for this macroblock mode is very similar to that of the motion-compensated macroblock modes with the only difference that the prediction signal (which is generated by motion-compensated prediction for the motion-compensated macroblock modes) is presented by the reconstruction of the macroblock that is obtained by decoding the base layer and all previous enhancement layers. For coding the residual (prediction error) signal of a macroblock, the residual coding as specified in the H.264/AVC standard including transformation, scaling, and quantisation is employed.

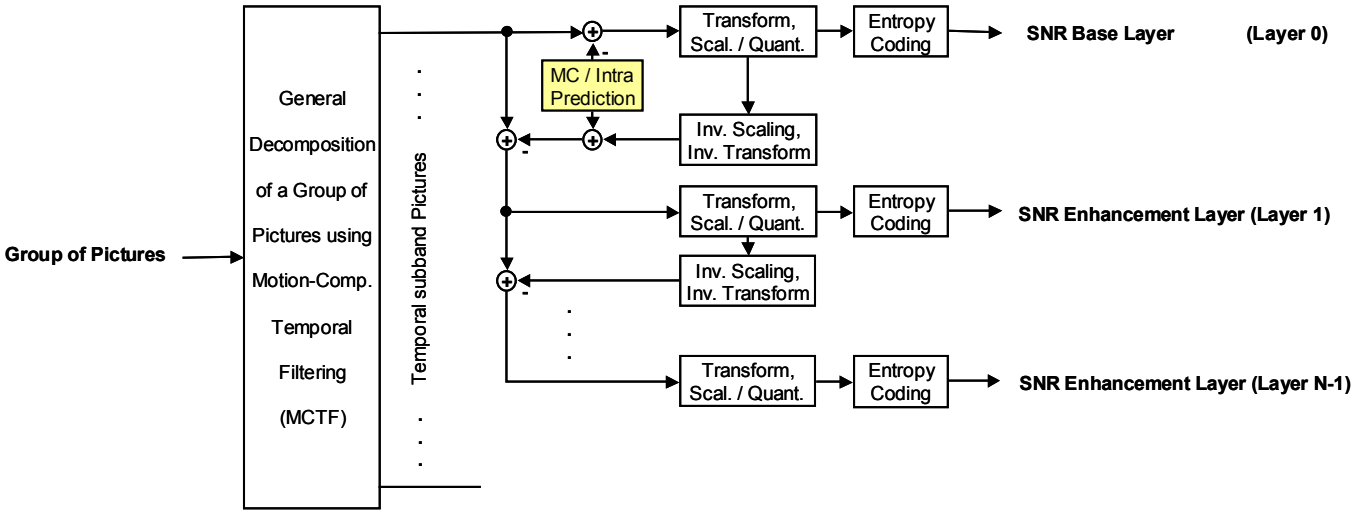

Figure 6: General concept of the SNR scalable coding scheme.

In the simplest version, the enhancement layer representations of the subband pictures are coded using a subset of the slice syntax of H.264/AVC, which is named **E slice** syntax. For each macroblock, only the residual information including the syntax elements `coded_block_pattern` and `mb_qp_delta` (if applicable) and the syntax element `end_of_slice_flag` are transmitted. Note, that the macroblock mode does not need to be coded, since in an E slice, all macroblocks are generally transmitted in the newly defined INTRA_BASE mode.

Especially for video sequences with high spatial detail and slow motion, the coding efficiency can further be improved, if for the enhancement layer representations of the low-pass pictures, motion-compensated prediction is allowed in addition the base layer prediction defined by the INTRA_BASE mode. This concept is illustrated in Figure 7. While the enhancement layer representations of high-pass pictures are exclusively predicted from the subordinate layer, macroblocks of the enhancement layer representations of low-pass pictures can be predicted from the subordinate layer or from previously reconstructed low-pass pictures of the same SNR layer, or they can be coded using the intra modes INTRA_4x4, INTRA_16x16, and INTRA_PCM defined in the standard. In Figure 7, base layer prediction is illustrated by dashed arrows, while solid arrows indicated motion-compensated prediction.

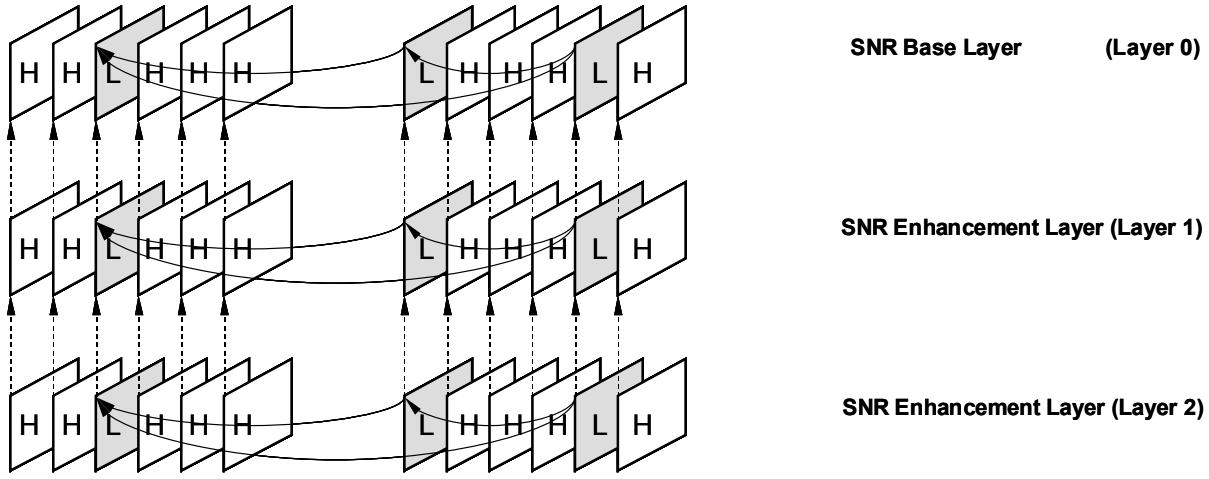

Figure 7: SNR scalable coding including motion compensated prediction for the enhancement representations of the low-pass signals.

Consequently, we additionally defined **IE**, **PE**, and **BE slices** for the coding of enhancement layer representations of low-pass pictures. The syntax of these IE, PE, and BE slices is similar to that of the standard I, P, and B slices, respectively. The only difference is that the newly defined INTRA_BASE macroblock mode is added to the sets of possible macroblock modes. Therefore, the codewords (or in case of CABAC, the binarisations and context variables) for transmitting the syntax element mb_type are slightly changed.

3.2.3 Spatial Scalability

In order to provide spatial scalability we adapted the spatial scalability concept as it is found in the video coding standards H.262/MPEG-2 Visual [7], H.263 [8], or MPEG-4 Visual [9] to the subband structure obtained by motion-compensated temporal filtering (MCTF). The spatial scalable coding scheme, which is very similar to the SNR scalable coding scheme presented in sec. 3.2.2, is illustrated in Figure 8 for one level of spatial scalability. A spatial base layer with reduced spatial resolution is coded using either the MCTF coding scheme presented in sec. 3.1 or standard H.264/AVC coding (in sec. 3.1 it was shown that standard H.264/AVC coding can be interpreted as a special case of the MCTF coding scheme). The reconstructed pictures $\{L^0\}$ of the base layer are spatially upsampled by a factor of k/m ($k > m > 1$), so that reconstructed pictures $\{L^{0*}\}$ with the same spatial resolution as the pictures of the next spatial enhancement layer are obtained. These upsampled pictures can be used for predicting the intra macroblocks in the subband pictures of the next spatial enhancement layer. It should be noted that in general not only the macroblocks of low-pass pictures, but also several macroblocks inside the high-pass pictures are coded in intra mode (cp. sec. 2.2). The reconstructed pictures $\{L^1\}$ of the first spatial enhancement layer can again by spatially upsampled and used as prediction signals for the intra macroblock of the next spatial enhancement layer. Thus, any level of spatial scalability can be provided.

The intra macroblocks that use the upsampled spatial base layer signal as prediction signal are coded in the INTRA_BASE macroblock mode, which was defined in sec. 3.2.2. And indeed, the reconstruction process for INTRA_BASE macroblocks is identical for SNR and spatial enhancement layers. The only differences between SNR and spatial enhancement layers are

- that in case of spatial scalability, the base layer signal is reconstructed using inverse motion-compensated filtering and upsampled before it is used as prediction signal for subsequent enhancement layers,
- that in case of SNR scalability, all macroblocks of the enhancement layer representations of high-pass pictures are inherently coded in INTRA_BASE mode, while for spatial enhancement layer representations of high-pass pictures the INTRA_BASE mode is only one possible macroblock mode for the intra macroblocks (intra macroblocks in a high-pass picture are specified by the corresponding prediction data array M_P), and
- that in spatial enhancement layers, the prediction data arrays M_P need to be transmitted in addition to the enhancement layer representations of the corresponding subband pictures.

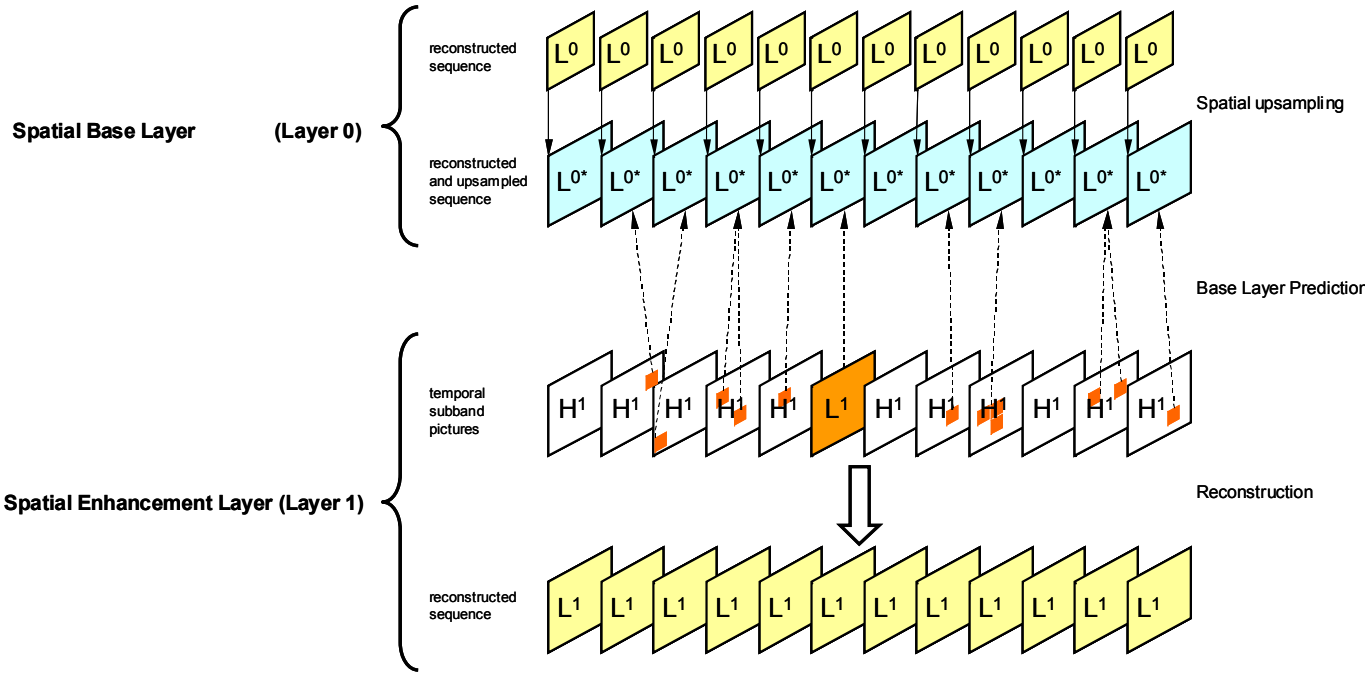

Figure 8: Concept for providing spatial scalability.

Consequently, the spatial enhancement layer representations of low-pass pictures are coded using the E, IE, PE, or BE slice syntax (or any combination of these) defined in sec. 3.2.2. For coding the spatial enhancement layer representations of high-pass pictures, an additional slice type named **HE** slice is defined. The syntax of the HE slice is similar to that of the H slice defined in sec. 3.1.2, with the only difference that the INTRA_BASE macroblock mode is added to the set of possible macroblock modes for the intra macroblocks of a high-pass picture (the intra macroblock of a high-pass picture are specified by the corresponding prediction data array M_P).

It is worth noting that with the presented concept the spatial resolution from on layer to the next can be increased by any factor k/m with $k > m > 1$. It is only necessary to define or transmit a set of corresponding interpolation filters. In our current implementation, we restrict ourselves to a spatial resolution conversion factor of 2, where the upsampling is performed with the 6-tap interpolation filter $\{1, -5, 20, 20, -5, 1\}$, which is defined in H.264/AVC for the purpose of half-sample interpolation.

3.2.4 Combined Scalability

The concepts of temporal, SNR, and spatial scalability presented in sec. 3.2.1, 3.2.2, and 3.2.3, respectively, can easily be combined to a general scalable coding scheme, which can provide a large amount of temporal, SNR, and spatial scalability. In Figure 9, an example for combined scalability is illustrated. This example is related to the Test Scenario 2 defined in the *Call for Proposals on Scalable Video Coding Technology* [10]. In this example, the spatial base layer (QCIF) is coded using standard H.264/AVC, where each second picture is transmitted as non-reference picture using the B slice syntax. Thus, for the spatial base layer one level of temporal scalability is provided. If only the reference pictures (I, P) are transmitted and decoded (Layer 0), a reconstructed sequence with a frame rate of 7.5 Hz is obtained. By additionally transmitting the non-reference pictures (B), the frame rate of the reconstructed sequence is increased to 15 Hz (Layer 1). It is easy to see that the same level of temporal scalability can be achieved, when the spatial base layer is coded using the more general MCTF coding scheme.

For coding the spatial enhancement layer with CIF resolution and a maximum frame rate of 30 Hz, we use the MCTF coding scheme with n decomposition stages. A representation of the third scalable layer (Layer 2: CIF, 15Hz, 256 kbit/s) is obtained, if in addition to the spatial base layer (Layer 0 and 1) first approximations of the low-pass picture(s) $\{L_{n-1}\}$ and the high-pass pictures $\{H_1\}$, …, $\{H_{n-1}\}$ are coded and the prediction data arrays $\{M_{P,1}\}$, …, $\{M_{P,n-1}\}$ are transmitted. For the next enhancement layer (Layer 3: CIF, 15Hz, 512 kbit/s), refinement signals for the low-pass picture(s) $\{L_{n-1}\}$ and the high-pass pictures $\{H_1\}$, …, $\{H_{n-1}\}$ are added. Layer 4 (CIF, 30Hz, 1024kbit/s) represents the next temporal and SNR enhancement layer; and thus, further refinements for the subband pictures $\{L_{n-1}\}$, $\{H_1\}$, …, $\{H_{n-1}\}$ and additionally first approximations of the high-pass pictures $\{H_0\}$ are coded and the prediction data

arrays $\{M_{P,0}\}$ are transmitted. For the last enhancement layer (Layer 5: CIF, 30Hz, 2048 kbit/s), which represents an SNR enhancement layer, refinement signals for all subband pictures $\{L_{n-1}\}$, $\{H_0\}$, …, $\{H_{n-1}\}$ are transmitted.

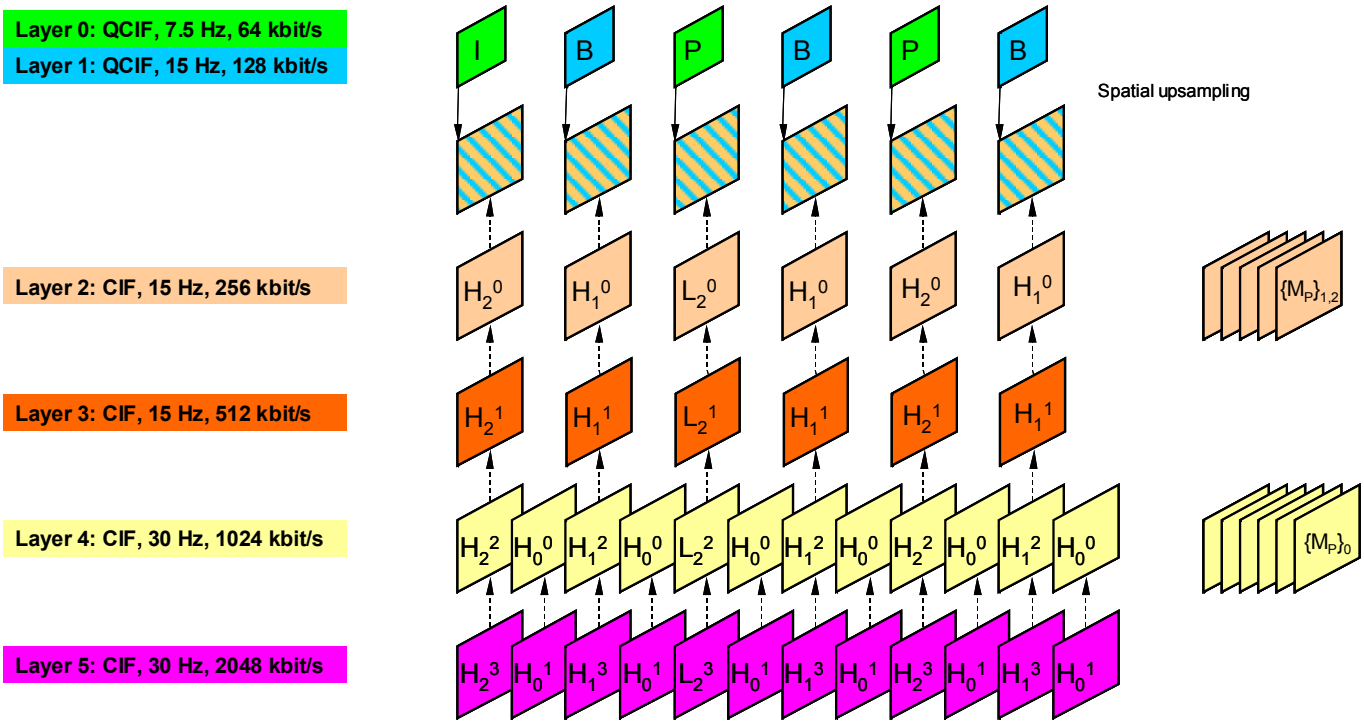

Figure 9: Example for combined scalability.

In the sec. 3.1.1 – 3.2.3, various slice types have been defined for the coding of base and enhancement layers with the MCTF coding scheme. In Table 1 and Table 2, the definition of the slice types supported by the general scalable MCTF scheme and their usage are summarized.

Finally, it should be pointed out again that the H.264/AVC standard could be interpreted as specialization of the presented coding scheme (cp. sec. 3.1) with groups of only 1 picture and a single spatial and SNR layer.

Table 1: Slice types and supported macroblock modes.

Slice Type	Supported macroblock modes					
	INTRA_4x4	INTRA_16x16	INTRA_PCM	INTRA_BASE	RESIDUAL	motion-compensated modes
M	X[(1)]					X
I	X	X	X			
P	X	X	X			X
B	X	X	X			X
IE	X	X	X	X		
PE	X	X	X	X		X
BE	X	X	X	X		X
E				X		
H	X	X	X		X[(2)]	
HE	X	X	X	X	X[(2)]	

(1) For M slices, the intra mode is called INTRA and it is not identical to the INTRA_4x4 mode (see sec. 2).

(2) The residual mode (RESIDUAL) is not indicated by the syntax element mb_type, instead the macroblocks that are coded in residual mode are specified by the corresponding prediction data array.

Table 2: Slice types and their usage.

Slice Type	Usage
M	Coding of prediction data arrays
I	Coding of base-layer (SNR, spatial) representations of low-pass pictures
P	
B	
IE	Coding of enhancement-layer (SNR, spatial) representations of low-pass pictures
PE	
BE	
E	Coding of SNR enhancement-layer representations of high-pass pictures Coding of enhancement-layer (SNR, spatial) representations of low-pass pictures
H	Coding of base-layer (SNR, spatial) representations of high-pass pictures
HE	Coding of spatial enhancement-layer representations of high-pass pictures

4 Operational Encoder Control

In this section, we described the operational encoder control that has been applied for generating the test bit streams, which we have submitted in response to the *Call for Proposals on Scalable Video Coding Technology* [10]. Sec. 4.1 explains the general coding and decomposition structure. In sec. 4.2 and 4.3, the algorithm for determining the prediction data arrays used for motion-compensated temporal filtering and the mode decision algorithm used for encoding the subband representation are described, respectively. The concept for selecting the quantisation parameters that are used for encoding the subband pictures is presented in sec. 4.4.

4.1 Scalability and Decomposition Structure

For generating the provided bit streams (for Test Scenario 2), we generally employed the coding structure depicted in Figure 9 and described in sec. 3.2.4. The QCIF layer (15 Hz) is coded using the standard H.264/AVC syntax. Every second picture is coded as B picture and marked as "not used for reference", while all other pictures are coded as P pictures (with exception of the first picture, which is coded as IDR picture) and marked as "used for reference". Thus, the QCIF layer provides the required level of temporal scalability; by only transmitting the reference pictures, a QCIF sequence with a frame rate of 7.5 Hz can be reconstructed.

For coding the CIF layer, the MCTF coding scheme including prediction from the upsampled QCIF layer is employed. The original CIF sequence with a frame rate of 30 Hz is partitioned into groups of 32 pictures, where the size of the last group of pictures is accordingly adapted. The groups of 32 pictures are decomposed into a single low-pass picture and 31 high-pass pictures using the algorithm described in sec. 2.3 with 5 decomposition stages and the control parameters depicted in Table 3. For the temporal motion-compensated filtering / decomposition, the number of active entries for all reference index lists *refIdxList0* and *refIdxList1* was set to 1 if the corresponding neighbouring picture exists, otherwise, the number of active entries was set to zero. Thus for motion-compensated prediction, only direct neighbouring pictures are used as reference pictures.

Table 3: Specification of the parameters *lowPassPartitioning*(*n*) and *skipUpdate*(*n*).

Decomposition stage *n*	***lowPassPartitioning*(*n*)**	***skipUpdate*(*n*)**
0	"10101010101010101010101010101010"	0
1	"1010101010101010"	0
2	"01010101"	0
3	"1010"	0
4	"01"	0

For encoding the low-pass pictures, we generally use the PE slice syntax (cp. sec. 3.2) with the exception of the low-pass picture of the first GOP, which is coded using the IE slice syntax. In the first CIF layer (LAYER 2: CIF, 15 Hz, 128 or 256 kbit/s), the representations of the high-pass pictures $\{H_1^0\}$, …, $\{H_{n-1}^0\}$ (cp. Figure 9) are transmitted using the HE slice syntax. For the following layers, the enhancement representations of the high-pass pictures are generally transmitted using the E slice syntax. However, in LAYER 4 (CIF, 30 Hz, 512 or 1024 kbit/s), the high-pass pictures $\{H_0^0\}$ (cp. Figure 9) that are not included in subordinate layers are coded using the H slice syntax. A detailed description of the properties and differences of the various slice types is given in sec. 3.2.

4.2 Motion estimation and mode decision process

This section describes an example method for determining the prediction data arrays M_P used in the prediction steps. This method was applied for generating the provided bit streams. The algorithm employs Lagrangian optimisation techniques that are widely used to optimise the rate-distortion efficiency of hybrid video coders [11]. A similar algorithm was integrated into the test model JM-2 [12] for the H.264/AVC standard.

Inputs to this process are a variable *idxLP*, reference index lists *refIdxList0* and *refIdxList1*, and an ordered set of low-pass pictures $\{ L_k[0], \ldots, L_k[N_k - 1] \}$.

Output of this process is a prediction data array M_P.

Furthermore, for controlling the motion estimation / mode decision process, the encoder control has to select the number of active entries for both reference index lists *refIdxList0* and *refIdxList1* as well as a quantisation parameter $QP \in [0; 51]$. The selected quantisation parameter determines the operating point of the encoder control.

Based on the given quantisation parameter QP, two Lagrangian multipliers λ_{SAD} and λ_{SSD} are derived by

$$\lambda_{SAD} = 0.92 \cdot 2^{QP/6-2}$$
$$\lambda_{SSD} = 0.85 \cdot 2^{QP/3-4}$$

Let R_0 and R_1 specify the set of active entries of the reference index lists *refIdxList0* and *refIdxList1*, respectively.

The prediction data array M_P is estimated in a macroblock-wise manner by using the following process.

1. For each possible macroblock partition P (and sub-macroblock mode $p_{sub\text{-}mb}$ if applicable) of the current macroblock, the prediction method p_{pred} together with the associated reference indices r_0 and/or r_1 and motion vectors $\{\mathbf{m}_0\}$ and $\{\mathbf{m}_1\}$ is determined by the following algorithm.

   - For all sub-macroblock partitions P_i of the current macroblock partition P (and sub-macroblock modes $p_{sub\text{-}mb}$ if applicable), list 0 and list 1 motion vector candidates $\mathbf{m}_0(r_0, i)$ and $\mathbf{m}_1(r_1, i)$ for all reference indices $r_0 \in R_0$ and $r_1 \in R_1$ are obtained by minimizing the Lagrangian functional

     $$\mathbf{m}_{0/1}(r_{0/1}) = \arg\min_{\mathbf{m}_{0/1} \in S} \{D_{SAD}(P_i, r_{0/1}, \mathbf{m}_{0/1}) + \lambda_{SAD} \cdot (R(r_{0/1}) + R(\mathbf{m}_{0/1}))\}$$

     with the distortion term being given as

     $$D_{SAD}(P, r_{0/1}, \mathbf{m}_{0/1}) = \sum_{(i,j) \in P} \left| l_{org}[i, j] - l_{ref,0/1}[i + m_{0/1,x}, j + m_{0/1,y}] \right|$$

     $l_{org}[]$ represents the luma sample array of the picture $L_k[idxLP]$, and $l_{ref,0/1}[]$ represents the luma sample array of the picture $L_k[refIdxList0/1[r_{0/1}]]$, which is referenced by the reference index $r_{0/1}$. S is the motion vector search range. The terms $R(r_{0/1})$ and $R(\mathbf{m}_{0/1})$ specify the number of bits needed to transmit the reference index $r_{0/1}$ and all components of the motion vector $\mathbf{m}_{0/1}$, respectively.

     The motion search proceeds first over all integer-sample accurate motion vectors in the given search range S. Then, given the best integer motion vector, the eight surrounding half-sample accurate motion vectors are tested, and finally, given the best half-sample accurate motion vector, the eight surrounding quarter-sample accurate motion vectors are tested. For the half- and quarter-sample

accurate motion vector refinement, the term $l_{\text{ref},0/1}[\, i + m_{0/1,\text{x}}\, , j + m_{0/1,\text{y}}\,]$ has to be interpreted as interpolation operator.

- Given the motion vector candidates for all sub-macroblock partitions P_i and reference indices $r_0 \in R_0$ and $r_1 \in R_1$, the list 0 and list 1 reference indices r_0 and r_1 and the associated motion vectors $\{\mathbf{m}_0\}$ and $\{\mathbf{m}_1\}$ for list 0 and list 1 prediction are selected by minimizing the Lagrangian functional

$$r_{0/1} = \arg\min_{r_{0/1} \in R_{0/1}} \left\{ \sum_{i \in P} \left(D_{SAD}(P_i, r_{0/1}, \mathbf{m}_{0/1}(r_{0/1}, i)) + \lambda_{SAD} \cdot R(\mathbf{m}_{0/1}(r_{0/1}, i)) \right) + \lambda_{SAD} \cdot R(r_{0/1}) \right\}$$

where the summation is proceeded over all sub-macroblock partitions P_i (with i being the sub-macroblock partition index) of a macroblock partition P.

Given the determined reference indices r_0 and r_1 and the associated motion vectors $\{\mathbf{m}_0\}$ and $\{\mathbf{m}_1\}$, the Lagrangian costs for list 0 and list 1 prediction J_{L0} and J_{L1} are calculated by

$$J_{\text{L0/L1}} = \sum_{i \in P} \left(D_{SAD}(P_i, r_{0/1}, \mathbf{m}_{0/1}(r_{0/1}, i)) + \lambda_{SAD} \cdot R(\mathbf{m}_{0/1}(r_{0/1}, i)) \right) + \lambda_{SAD} \cdot R(r_{0/1})$$

- Given the reference indices r_0 and r_1 and the associated motion vectors $\{\mathbf{m}_0\}$ and $\{\mathbf{m}_1\}$ for list 0 and list 1 prediction, the reference indices r_{B0} and r_{B1} and the associated motion vectors $\{\mathbf{m}_{\text{B0}}\}$ and $\{\mathbf{m}_{\text{B1}}\}$ for bi-prediction are obtained by the following iterative algorithm.

  Initially, the reference indices and motion vectors for bi-prediction are set equal to the reference indices and motion vectors that have been determined for list 0 and list 1 prediction,

  $r_{\text{B0}} = r_0, \qquad \mathbf{m}_{\text{B0}} = \mathbf{m}_0,$
  $r_{\text{B1}} = r_1, \qquad \mathbf{m}_{\text{B1}} = \mathbf{m}_1,$

  an iteration index *iter* is set equal to 0,

  $iter = 0$

  and the Lagrangian cost for bi-prediction J_{BI} is set equal to $J(\, r_{\text{B0}}, r_{\text{B1}}, \{\mathbf{m}_{\text{B0}}\}, \{\mathbf{m}_{\text{B1}}\}\,)$ with

$$J(r_0, r_1, \{\mathbf{m}_0\}, \{\mathbf{m}_1\}) =$$
$$\left\{ \sum_{i \in P} \left(D_{SAD}(P_i, r_0, r_1, \mathbf{m}_0(i), \mathbf{m}_1(i)) + \lambda_{SAD} \cdot \left(R(\mathbf{m}_0(i)) + R(\mathbf{m}_1(i)) \right) \right) + \lambda_{SAD} \cdot \left(R(r_0) + R(r_1) \right) \right\}$$

  and the distortion term being given as

$$D_{SAD}(P, r_0, r_1, \mathbf{m}_0, \mathbf{m}_1) = \sum_{(i,j) \in P} \left| l_{org}[i, j] - \left(l_{ref,0}[i + m_{0x}, j + m_{0y}] + l_{ref,1}[i + m_{1x}, j + m_{1y}] + 1 \right) / 2 \right|$$

  Subsequently, in each iteration step, the following applies.

  - If (iter % 2) is equal to 0, the following applies.
    - A list 0 reference index r^*_{B0} and associated list 0 motion vectors $\{\mathbf{m}^*_{\text{B0}}\}$ are determined by minimizing the following Lagrangian functional

$$r^*_{B0} = \arg\min_{r_0 \in R_0} \left\{ \sum_{i \in P} \min_{\mathbf{m}_0 \in S^*(\mathbf{m}_{B0})} \left(D_{SAD}(P_i, r_0, r_{B1}, \mathbf{m}_0(i), \mathbf{m}_{\text{B1}}(i)) + \lambda_{SAD} \cdot R(\mathbf{m}_0(i)) \right) + \lambda_{SAD} \cdot R(r_0) \right\}$$

      where the search range $S^*(\mathbf{m}_{\text{B0}}(\text{i}))$ specifies a small area around the motion vector $\mathbf{m}_{\text{B0}}(\text{i})$.

    - If the associated cost measure $J_{\text{iter}} = J(\, r^*_{\text{B0}}, r_{\text{B1}}, \{\mathbf{m}^*_{\text{B0}}\}, \{\mathbf{m}_{\text{B1}}\}\,)$ is less than the minimum cost measure J_{BI}, the list 0 reference index r^*_{B0} is assigned to r_{B0} and the associated list 0 motion vectors $\{\mathbf{m}^*_{\text{B0}}\}$ are assigned to $\{\mathbf{m}_{\text{B0}}\}$.
  - Otherwise ((iter % 2) is equal to 1), the following applies.
    - A list 1 reference index r^*_{B1} and associated list 1 motion vectors $\{\mathbf{m}^*_{\text{B1}}\}$ are determined by minimizing the following Lagrangian functional

$$r_{B1}^{*} = \arg\min_{r_1 \in R_1} \left\{ \sum_{i \in P} \min_{\mathbf{m}_1 \in S^{*}(\mathbf{m}_{B1})} \left(D_{SAD}(P_i, r_{B0}, r_1, \mathbf{m}_{B0}(i), \mathbf{m}_1(i)) + \lambda_{SAD} \cdot R(\mathbf{m}_1(i)) \right) + \lambda_{SAD} \cdot R(r_1) \right\}$$

- If the associated cost measure $J_{\text{iter}} = J(\, r_{\text{B0}}, r^{*}_{\text{B1}}, \{\mathbf{m}_{\text{B0}}\}, \{\mathbf{m}^{*}_{\text{B1}}\}\,)$ is less than the minimum cost measure J_{BI}, the list 1 reference index r^{*}_{B1} is assigned to r_{B1} and the associated list 1 motion vectors $\{\mathbf{m}^{*}_{\text{B1}}\}$ are assigned to $\{\mathbf{m}_{\text{B1}}\}$.

o If the calculated cost measure J_{iter} is greater than or equal to the minimum cost measure J_{BI}, the iteration process is stopped.

o Otherwise, J_{BI} is set equal to J_{iter}.

o If a maximum number of iterations has been carried out, the iteration process is stopped.

o Otherwise, the iteration index *iter* is incremented by 1: *iter* = *iter* + 1.

- Given the reference indices r_0, r_1, r_{B0}, and r_{B1} and the associate motion vectors $\{\mathbf{m}_0\}$, $\{\mathbf{m}_1\}$, $\{\mathbf{m}_{\text{B0}}\}$, and $\{\mathbf{m}_{\text{B1}}\}$ as well as the Lagrangian cost measures J_{L0}, J_{L1}, and J_{BI}, the prediction method $p_{\text{pred}} \in \{\text{ L0, L1, Bi }\}$ for the current macroblock partition P (and sub-macroblock mode $p_{\text{sub-mb}}$ if applicable), and thus the associated reference indices and motion vectors, is chosen by

$$p_{pred} = \left(\left(J_{\text{BI}} \le J_{\text{L0}}\right) \&\& \left(J_{\text{BI}} \le J_{\text{L1}}\right) ? \text{ BI} : \left(\left(J_{\text{L0}} \le J_{\text{L1}}\right) ? \text{ L0} : \text{L1} \right) \right)$$

2. For all sub-macroblocks (8x8 luma blocks) $P_{\text{sub-mb}}$, the sub-macroblock mode $p_{\text{sub-mb}}$ is determined by minimizing the Lagrangian functional

$$p_{sub-mb} = \arg\min_{p \in S_{sub-mb}} \left\{ D_{SSD}(P_{sub-mb}, p) + \lambda_{SSD} \cdot R(p) \right\}$$

with the distortion term

$$D_{SSD,Y}(P, p) = \sum_{(i,j) \in P} \left(l_{org}[i,j] - l_{rec}(p)[i,j] \right)^2$$

$l_{\text{rec}}(p)[]$ represents the arrays of reconstructed luma samples that are obtained after applying

- motion-compensated prediction using the prediction method p_{pred} together with the reference indices and motion vectors that have been determined for the sub-macroblock mode p,
- transformation, scaling, and quantisation (with the given quantisation parameter QP) of the luma component of the prediction residual,
- inverse scaling, and inverse transformation for the luma component of the prediction residual.

$S_{\text{sub-mb}}$ represents the set of possible sub-macroblock modes, which is given by { B_Direct_8x8, B_XX_8x8, B_XX_8x4, B_XX_4x8, B_XX_4x4 } with the template XX being replaced by the chosen prediction method $p_{\text{pred}} \in \{\text{ L0, L1, Bi }\}$. $R(p)$ specifies the number of bits that is needed for transmitting the sub-macroblock mode (including the prediction method), the associated reference indices and motion vectors as well as the quantized luma residual signal.

3. Similarly to step 2, the intra prediction modes p_{ipred} for the 4x4 luma blocks (for the INTRA_4x4 macroblock mode), the 16x16 luma blocks (for the INTRA_16x16 macroblock mode), and the 8x8 chroma blocks (for the INTRA_4x4 and the INTRA_16x16 macroblock mode) are determined by minimizing

$$p_{ipred} = \arg\min_{p \in S_{ipred}} \left\{ D_{SSD,Y/C}(P, p) + \lambda_{SSD} \cdot R(p) \right\}$$

with S_{ipred} being the set of possible intra prediction modes. P represents the area of the regarded 4x4 luma block, 16x16 luma block, or 8x8 chroma block. $R(p)$ specifies the number of bits that is needed for transmitting the intra prediction mode and the quantized luma or chroma residual signal. The reconstructed sample values are obtained by intra prediction and subsequent transformation, scaling, quantisation (with QP), inverse scaling, and inverse transformation of the residual signal.

For the determination of chroma intra prediction modes, the term $D_{SSD,Y}(\ldots)$ specifying the distortion of the luma component is replaced by

$$D_{SSD,C}(P, p) = \sum_{(i,j)\in P} \left(u_{org}[i,j] - u_{rec}(p)[i,j]\right)^2 + \left(v_{org}[i,j] - v_{rec}(p)[i,j]\right)^2$$

specifying the distortion of the chroma components. $u_{org}[]$, $v_{org}[]$ and $u_{rec}[]$, $v_{rec}[]$ represent the original and reconstructed sample arrays of the chroma components of the picture $L_k[idxLP]$, respectively.

4. Finally, the macroblock mode p_{mb} is obtained by minimizing the Lagrangian functional

$$p_{mb} = \arg\min_{p\in S_{mb}} \left\{D_{SSD,Y}(P_{MB}, p) + D_{SSD,C}(P_{MB}, p) + \lambda_{SSD} \cdot R(p)\right\}$$

P_{MB} represents the current macroblock. S_{mb} is the set of possible macroblock modes given by { B_Direct_16x16, B_XX_16x16, B_XX_YY_16x8, B_XX_YY_8x16, B_8x8, INTRA_4x4, INTRA_16x16, INTRA_PCM } with the templates XX and YY being replaced by the chosen prediction method $p_{pred} \in$ { L0, L1, Bi } for the first and second macroblock partition, respectively. If the current spatial layer constitutes a spatial enhancement layer and a prediction picture exists for the current picture $L_k[idxLP]$, the INTRA_BASE mode is added to the set of potential macroblock modes. The term $R(p)$ specifies the number of bits needed for transmitting all syntax elements that are required for reconstructing the macroblock including the macroblock mode, the sub-macroblock modes (if applicable), the intra prediction modes (if applicable), the reference indices and motion vectors (if applicable), and the quantized residual signal (luma and chroma components).

5. In case, the selected macroblock mode is an intra mode (INTRA_4x4, INTRA_16x16, INTRA_PCM, or INTRA_BASE), the macroblock mode p_{mb} is set to INTRA.

4.3 Coding of Subband Pictures

In principle, the algorithm for determining the coding modes (macroblock and sub-macroblocks modes) and associates parameters for the encoding of low- and high-pass pictures (base layers) or corresponding refinement signals (enhancement layers) is identical to the algorithm presented in sec. 4.2. The following differences exist:

- The set of possible macroblock modes is dependent on the selected slice type. In Table 1, an overview of the supported macroblock modes for each slice type is given.
- For H and HE slices, the macroblock mode decision process is only applied for intra macroblocks, which are specified by the corresponding prediction data array M_P; the residual macroblock mode (RESIDUAL) is never included in the set of possible macroblock modes.
- Since for E slices the macroblock mode is always set to INTRA_BASE, no mode decision process is performed for this slice type.
- For I, IE, H, and HE slices, the steps 1 and 2 of the algorithm presented in sec. 4.2 are not executed.
- Step 5 of the algorithm presented in sec. 4.2 is not executed.

The quantisation of the scaled transform coefficients of the residual signals is performed as specified in the H.264/AVC Test Model JM-2 [12].

4.4 Quantizer Selection

When neglecting the motion and replacing the bit-shift to the right in the update step by a real-valued multiplication by a factor of 1/2, the basic two-channel analysis step (see sec. 2) can be normalized by multiplying the high-pass samples of the picture H by a factor of 1/sqrt(2) (in case of uni-directional prediction) or sqrt(2/3) (in case of bi- prediction) and the low-pass samples by a factor of sqrt(2) (in case of uni-directional prediction) or sqrt(32/23) (in case of bi- prediction). Since we neglect this normalization in the realization of the analysis and synthesis filter banks to keep the range of the samples values nearly constant, we have to take it into account during the quantisation of the temporal subbands.

Let QP_{r0} be a given real-valued quantisation parameter. For each layer and each group of pictures this quantisation parameter has to be determined by the encoder control.

The quantisation parameters $QP_H(k, i)$ and $QP_L(k, i)$ that are used for encoding the low- and high-pass pictures $L_k[i]$ and $H_k[i]$ (or their refinement representations), which are obtained after the k-th decomposition stage ($k = 0,\ldots, n - 1$), are set to

$$QP_L(k, i) = \min(51, \max(0, \text{Round}(QP_{rL}(k, i)))),$$

$$QP_H(k, i) = \min(51, \max(0, \text{Round}(QP_{rH}(k, i)))).$$

At this, n is the number of decomposition stages that have been applied to the considered group of pictures. The operator Round() specifies rounding to the nearest integral number. $QP_{rL}(k, i)$ and $QP_{rH}(k, i)$ represent real-valued approximations of the quantisation parameters $QP_H(k, i)$ and $QP_L(k, i)$, and are calculated as follows

$$QP_{rL}(k, i) = QP_{pred,L}(k, i) - \Delta QP_L(k, i),$$

$$QP_{rH}(k, i) = QP_{pred,H}(k, i) + \Delta QP_H(k, i),$$

where $QP_{pred,L}(k, i)$ and $QP_{pred,H}(k, i)$ represent predictions for the real-valued quantisation parameters $QP_{rL}(k, i)$ and $QP_{rH}(k, i)$. These quantisation parameter predictions $QP_{pred,L}(k, i)$ and $QP_{pred,H}(k, i)$ are determined as follows.

- If k is equal to 0, $QP_{pred,L}(k, i)$ and $QP_{pred,H}(k, i)$ are set equal to the given quantisation parameter QP_0.
- Otherwise (k is greater than 0), $QP_{pred,L}(k, i)$ and $QP_{pred,H}(k, i)$ are set to the average of the quantisation parameters $\{ QP_{rL}(k-1, ...) \}$ of the low-pass picture $L_{k-1}[j]$, which shares the coordinate system with the regarded low- or high-pass picture $L_k[i]$ or $H_k[i]$, and its direct neighbours (if they exist).

The real-valued quantisation parameter differences $\Delta QP_H(k, i)$ and $\Delta QP_L(k, i)$ are determined as follows

$$\Delta QP_L(k, i) = 3 \cdot ((N_{Bi}(k, i) / N_{tot}) \cdot \log_2(32 / 23) + (N_{Uni}(k, i) / N_{tot}))$$

$$\Delta QP_H(k, i) = 3 \cdot ((N_{Bi}(k, i) / N_{tot}) \cdot \log_2(3 / 2) + (N_{Uni}(k, i) / N_{tot}))$$

N_{tot} represents the total number of luma samples inside a picture. $N_{Bi}(k, i)$ and $N_{Uni}(k, i)$ specify the number of samples in the regarded low- or high-pass picture $L_k[i]$ or $H_k[i]$ that can be classified as bi-directional and uni-directional connected, respectively. The number of bi-directional and uni-directional connected samples $N_{Bi}(k, i)$ and $N_{Uni}(k, i)$ is determined by analysing the prediction data arrays $\{M_P\}$ and $\{M_U\}$ that are used in the corresponding prediction and update steps. The actual derivation process is similar to the derivation process for the prediction data arrays M_U used in the update steps (cp. sec. 2.3.5).

Within each temporal subband picture, the quantisation parameter QP is held constant in our current encoder implementation, which was used for generating the submitted bit streams. Note, that using the described algorithm, the bit-allocation process (and thus the rate as well as distortion) for each layer and each group of pictures is determined by a single real-valued quantisation parameter QP_0.

5 Simulation Results

The coding efficiency of the proposed scalable extension of H.264/AVC is evaluated for the coding conditions and test sequences that are specified for Test Scenario 2 in the *Call for Proposals on Scalable Video Coding Technology* [10]. The rate-distortion performance of various version of the scalable H.264/AVC extension is compared to the rate-distortion performance of the non-scalable H.264/AVC anchor bit streams that have been provided by MPEG [14].

The rate-distortion plots for the four test sequences *Bus*, *Foreman*, *Football*, and *Mobile* are depicted in Figure 10 – Figure 13. Beside the rate-distortion curve for the non-scalable H.264/AVC compliant anchor bit streams (*black curve*), rate-distortion curves for the following versions of the scalable coding scheme are plotted into the diagrams:

1. The scalable coding scheme (with 6 scalability layers) described in sec. 4.1 (*solid red curve*). This configuration was used for generating the bit streams that have been submitted in response to the *Call for Proposals on Scalable Video Coding Technology* [10].

2. The scalable coding scheme of point 1 with the difference that the QCIF layer is not used for predicting intra macroblocks in the CIF layer (*dashed red curve*). The QCIF and CIF layer are transmitted via simulcast. In the rate-distortion plots, the bit rate for the CIF layers is calculated as the sum of the bit rates for the actual CIF layer and the QCIF layer (with a frame rate of 15 Hz). Note, that the QCIF layer is identical to the QCIF layer of the scalable coding scheme of point 1.
3. The scalable coding scheme of point 1 with the difference that only the CIF layers are transmitted (*blue curve*). Thus, only four scalability levels are provided.
4. A non-scalable version of the coding scheme using motion-compensated temporal filtering (*green curve*). For this encoder configuration, the same temporal decomposition structure as for the scalable coding scheme of point 1 was used. However, the subband pictures and prediction data arrays are coded using the single layer approach presented in sec. 3.1; only the CIF versions of the test sequences have been encoded.
5. A low-delay version of the scalable coding scheme of point 1 (*pink curve*). In order to achieve a total encoding-decoding delay less than 150 ms, the CIF versions of the test sequences with full temporal resolution (30 Hz) are coded in groups of only 4 pictures. In this version, the QCIF layers are also coded using the general MCTF coding scheme; the temporal decomposition of the QCIF versions of the test sequences (15 Hz) is accordingly applied to groups of only 2 pictures.

While the encoders of the points 1, 2, 3, and 5 generate embedded bit streams from which the various spatial and temporal resolutions and bit rates can be extracted, the H.264/AVC compliant encoder and the encoder of point 4 generate non-scalable bit streams, i.e. for each rate-distortion point plotted in the diagrams a different bit stream has been encoded. All encoders including the H.264/AVC compliant encoder use a similar degree of encoder optimisation; in fact, nearly the same encoder control is applied.

By analysing the rate-distortion diagrams depicted in Figure 10 – Figure 13, the following observations can be made.

- The coding efficiency of the scalable coding scheme with 6 scalability layers (point 1, *solid red curve*) is between 0 to 3 dB worse than that of the non-scalable H.264/AVC compliant encoder (*black curve*). The largest SNR differences are observed for the lowest-rate CIF layers. This is related to the fact that for the lowest-rate CIF layer, a large part of the available bit-rate is needed for transmitting the QCIF base layers and the prediction information. Thus, for our proposed coding scheme, a bit-rate ratio between the lowest-rate CIF layer and the highest-rate QCIF layer of 2:1 as it is defined in the Test Scenario 2, is unfavourable. For the QCIF layers and the CIF layers with a frame rate of 30 Hz, the SNR difference between the H.264/AVC compliant encoder and our scalable coding scheme is always less than 1.5 dB. For the sequences *Foreman* and *Mobile*, the SNR's of the CIF layers extracted from the embedded scalable bit stream are even slightly higher than the SNR's for the non-scalable H.264/AVC compliant anchor bit streams.
- The coding efficiency of the scalable encoder using simulcast (point 2, *dashed red curve*) is significantly worse than the coding efficiency of the scalable encoder employing prediction from the QCIF layer (point 1, *solid red curve*) for the lowest-rate CIF layers. This observation verifies the design of the scalable coding scheme with respect to the spatial scalability.
- By comparing the rate-distortion efficiency of the scalable coding scheme with 6 scalability layers (point 1, *solid red curve*) with the rate-distortion efficiency of the scalable coding scheme with only 4 scalability layers (point 3, *blue curve*), it can be seen that the coding efficiency is improved if only the four CIF layers are coded. This is mainly related to the above-mentioned fact that a large part of the bit-rate of the lowest-rate CIF layer is needed for transmitting the QCIF base layer and the prediction information. In order to meet the bit-rate constraints, it is often necessary to enlarge the Lagrangian parameter that controls the prediction data bit rate (see sec. 4.2) in a way that a clearly sub-optimal temporal decomposition for the high-rate layers is obtained.
- By using a non-scalable version of the presented MCTF coding scheme (point 4, *green curve*), the coding efficiency is further improved. Again, this is mainly related to the fact that the motion estimation / mode decision process for determining the prediction information can only be optimised for a single operating point. For non-scalable coding the selection of the operating point can be

adjusted to the target bit rate. However, for scalable coding, the operating point has to be defined anywhere inside the supported bit-rate interval; and thus, the determined prediction data arrays used for temporal decomposition are sub-optimal for most of the scalability layers. Nevertheless, the rate-distortion performance of the non-scalable MCTF coding scheme clearly demonstrates the potential of motion-compensated filtering as a tool for increasing the coding efficiency. Indeed, for three of the four test sequences (with exception of the *Football* sequence), the non-scalable MCTF coding scheme outperforms the H.264/AVC compliant encoder. SNR gains of up to 1.5 dB have been observed, and we could also observe an improvement in subjective quality for the test sequences in CIF resolution with a frame rate of 30 Hz.

- Finally, the scalable coding scheme that was used for generating the provided bit streams (point 1, *solid red curve*) is compared to a low-delay version (point 5, *pink curve*) providing the same degree of scalability. By introducing a maximum encoding-decoding delay of 150 ms, the coding efficiency is decreased by 0 to about 1 dB for the test sequences *Bus*, *Foreman*, and *Football*. For the sequence *Mobile*, a drop in SNR of up to 2 dB has been observed.

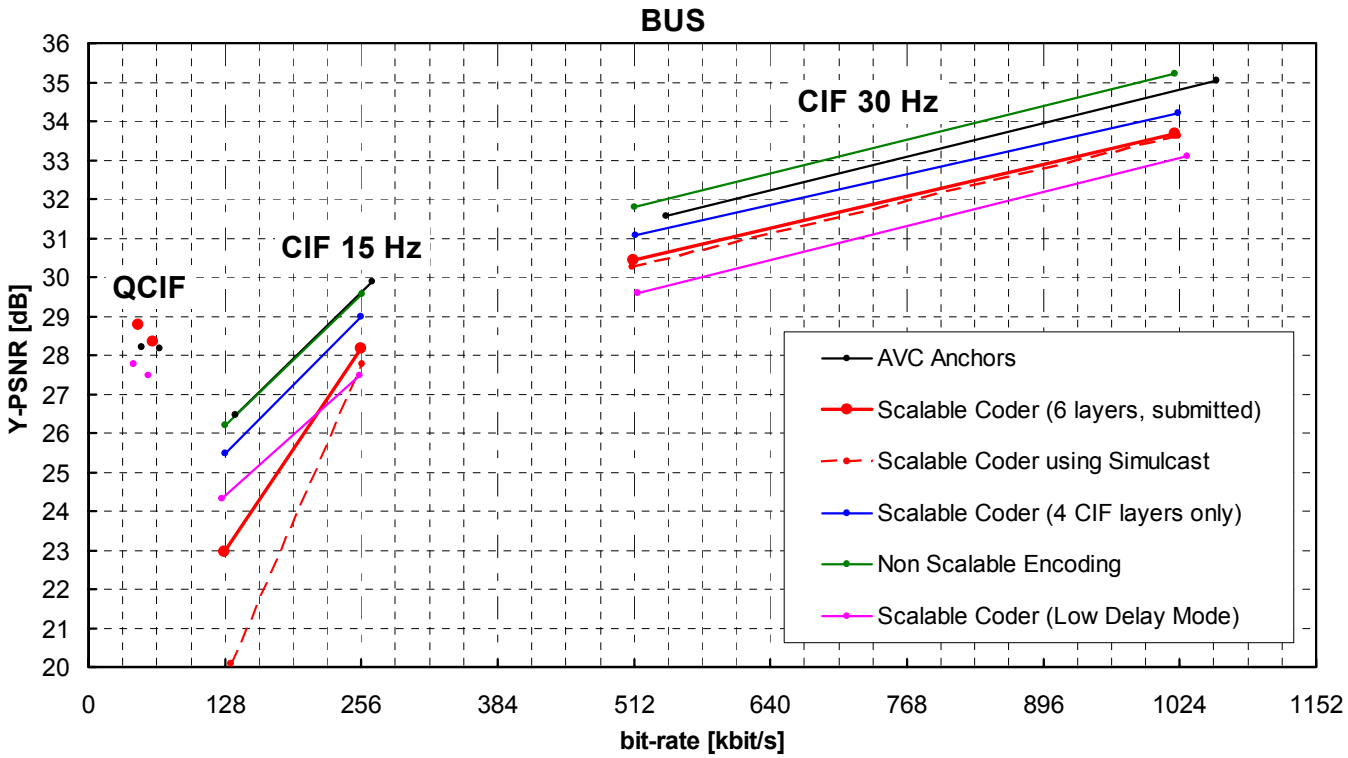

Figure 10: Comparison of the coding efficiency of an H.264/AVC compliant encoder and various versions of the proposed scalable extension for the sequence *Bus*. The red solid curve shows that rate-distortion performance of the embedded bit stream that we have submitted in response to the *Call for Proposals on Scalable Video Coding Technology* [10].

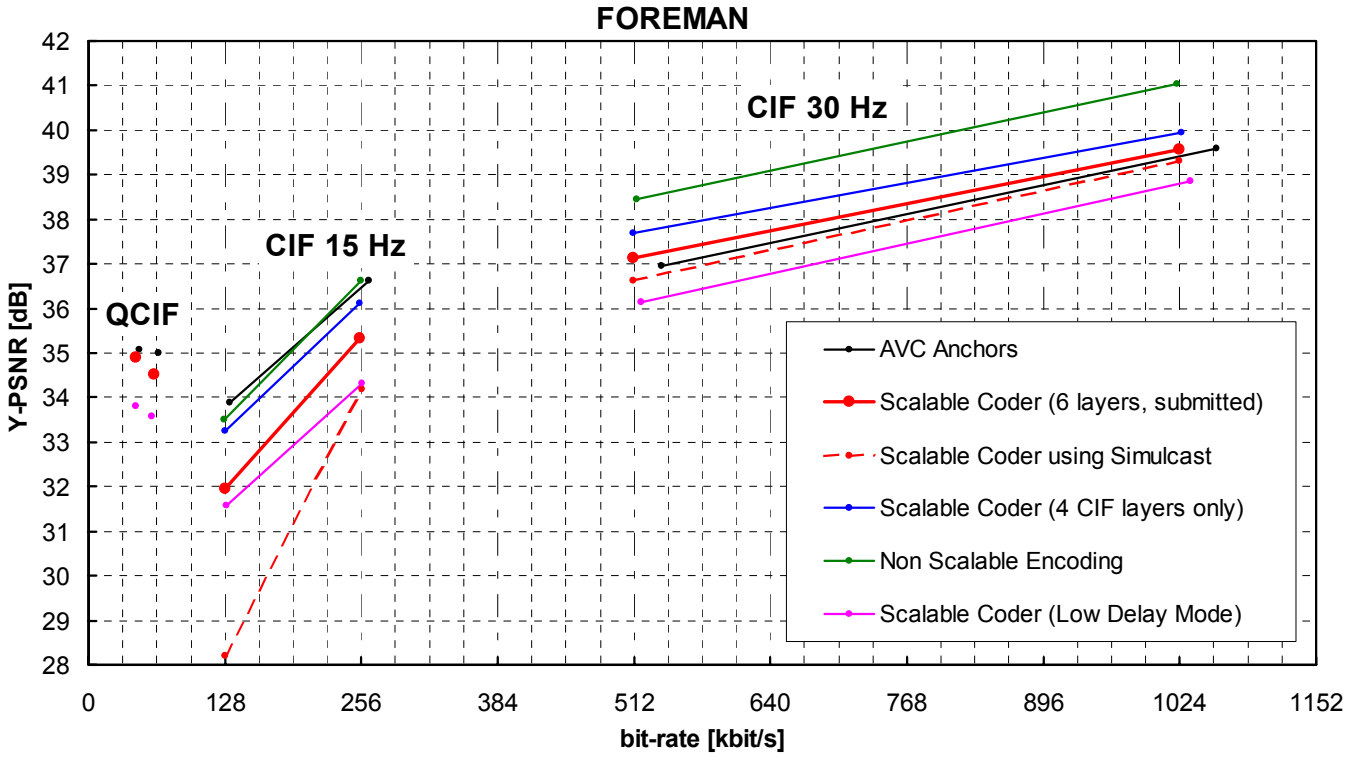

Figure 11: Comparison of the coding efficiency of an H.264/AVC compliant encoder and various versions of the proposed scalable extension for the sequence *Foreman*. The red solid curve shows that rate-distortion performance of the embedded bit stream that we have submitted in response to the *Call for Proposals on Scalable Video Coding Technology* [10].

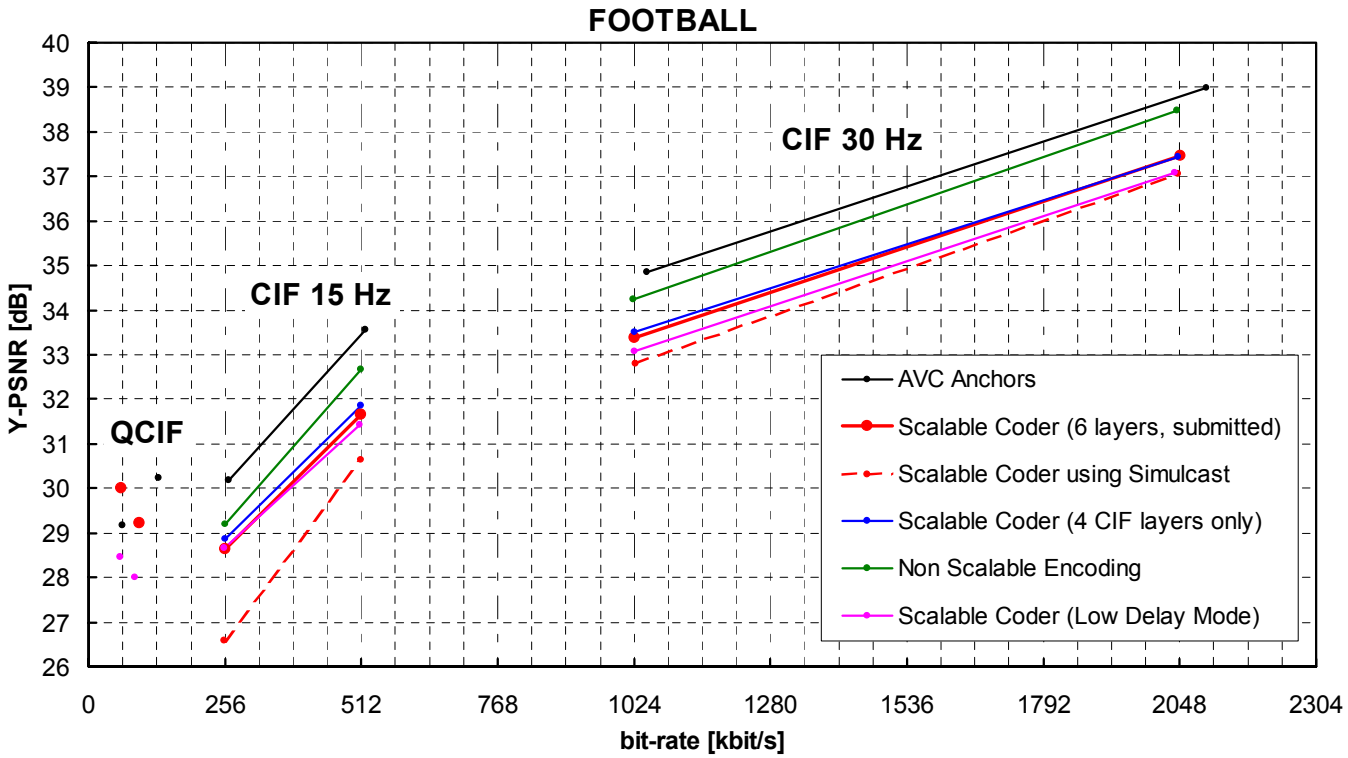

Figure 12: Comparison of the coding efficiency of an H.264/AVC compliant encoder and various versions of the proposed scalable extension for the sequence *Football*. The red solid curve shows that rate-distortion performance of the embedded bit stream that we have submitted in response to the *Call for Proposals on Scalable Video Coding Technology* [10].

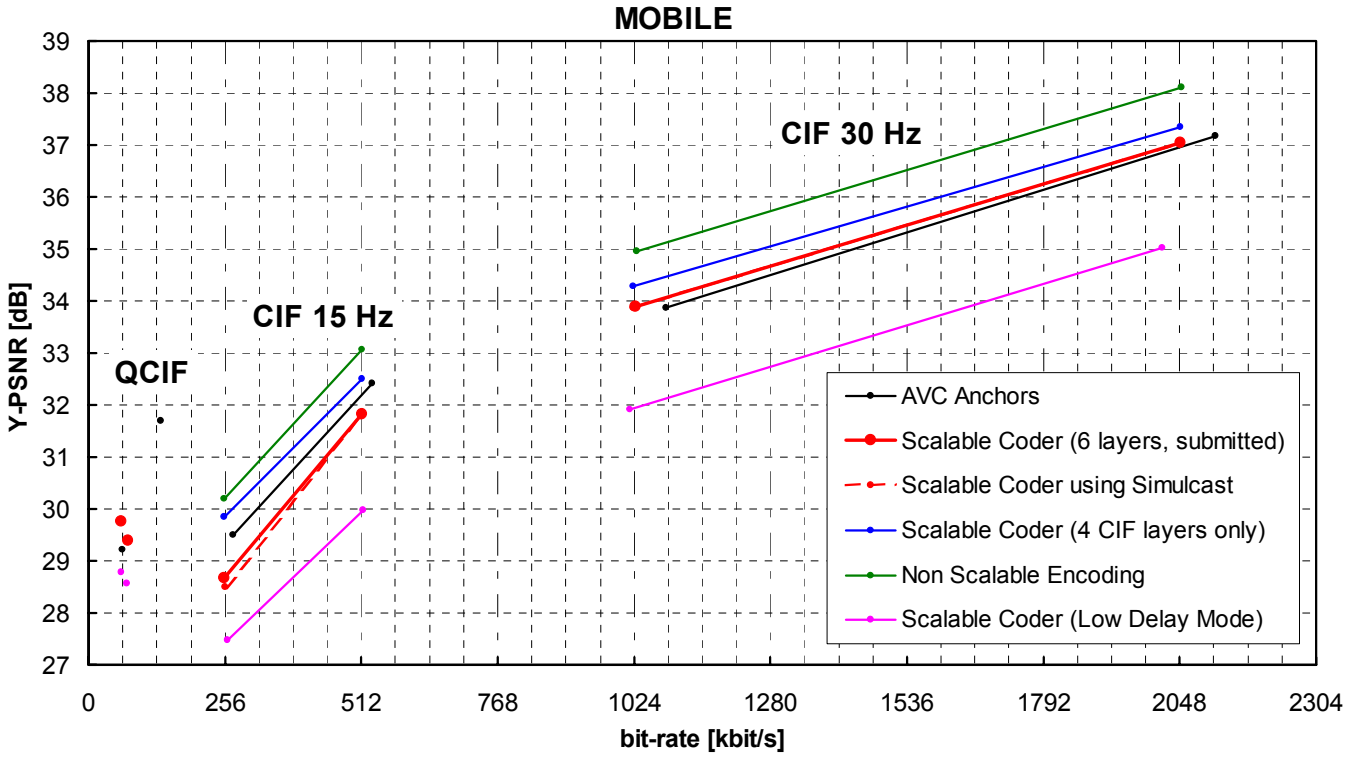

Figure 13: Comparison of the coding efficiency of an H.264/AVC compliant encoder and various versions of the proposed scalable extension for the sequence *Mobile*. The red solid curve shows that rate-distortion performance of the embedded bit stream that we have submitted in response to the *Call for Proposals on Scalable Video Coding Technology* [10].

6 Fulfillment of Requirements and Additional Information

6.1 Fulfillment of Requirements

The Table 4 contains a point-by-point assessment of how the proposed coding scheme satisfies the requirements on Scalable Video Coding defined by MPEG [13].

Table 4: Fulfillment of requirements on Scalable Video Coding defined in [13].

Requirement		Comments
1	**Spatial scalability**	In principle, the proposed coding scheme supports an **arbitrary number of spatial scalability levels**. The spatial resolution from one spatial scalability level to the next can be increased by any factor k / m with k > m > 1 (cp. sec. 3.2.3).
2	**Temporal scalability**	In principle, the proposed coding scheme supports an **arbitrary number of temporal scalability levels**. For a specific application, the number of supported temporal scalability levels is only restricted by the corresponding delay and/or memory constraints. The temporal resolution from one temporal scalability level to the next can be increased by any factor k / m with k > m > 1. For applications that require a constant sampling period between successive reconstructed pictures, the temporal resolution from one layer to the next can be increased by any integral number k > 0 (cp. sec. 3.2.1).
3	**SNR scalability**	The proposed coding scheme supports an **arbitrary number of quality (SNR) scalability levels** (cp. sec. 3.2.2). The bit rate of the SNR enhancement layers, and thus the granularity of the SNR scalability can arbitrarily be adjusted.

4	**Complexity scalability**	The proposed coding scheme **supports a mechanism that enables complexity scalability**. The complexity of the decoding process is determined by the number of spatial, temporal, and quality (SNR) layers that are used for decoding. In addition, the complexity of the reconstruction process for a group of pictures can further be reduced by omitting several composition stages (whereby the temporal resolution is further decreased) or by skipping the update steps and/or the deblocking filter process.
5	**Region of interest scalability**	In the proposed coding scheme, a **region of interest scalability can be enabled** by using the feature "*Flexible Macroblock Ordering*" supported in the H.264/AVC standard [1] (num_slice_groups_minus1 > 0 with slice_group_map_type = = 6). It is also possible to provide region of interest scalability by a suitable adjustment of the macroblock quantisation parameters in different SNR layers.
6	Object based scalability	Object-based scalability is not supported in the proposed coding scheme.
7	**Combined scalability**	The proposed coding scheme supports combined spatial, temporal, and quality (SNR) scalability (cp. sec. 3.2.4). The complexity of the decoding process is determined by the number of spatial, temporal, and quality (SNR) layers that are used for decoding. In addition, the complexity of the reconstruction process for a group of pictures can further be reduced by omitting several composition stages (whereby the temporal resolution is further decreased) or by skipping the update steps and/or the deblocking filter process (cp. *Complexity Scalability*). Thus, **a combined spatial, temporal, quality (SNR), and complexity scalability is supported**. The spatial and/or temporal resolution from one layer to the next can be increased by any factor k / m with k > m > 1 (cp. *Spatial Scalability* and *Temporal Scalability*). For the quality (SNR) scalability, the granularity can arbitrarily be adjusted (see *SNR Scalability*).
8	**Robustness to different types of transmission errors**	Due to the open-loop structure of the proposed coding scheme, **it is possible to efficiently incorporate robustness to different types of transmission errors**. The following features/methods can be used to increase the robustness to transmission errors: • Unequal error protection with increased error protection for the NAL units of the base layer (and subordinate enhancement layers) • Slice data partitioning (similar to H.264/AVC [1]) for the high-pass frames with one data partition for the intra and another for the residual macroblocks in connection with unequal error protection, where increased error protection should be used for the NAL units containing parameter sets (if not delivered out of band), slices of the low-pass pictures, slices of prediction data arrays, and slice data partitions with intra macroblocks of the high-pass pictures. • Redundant slices (as specified in H.264/AVC [1]) • Flexible macroblock ordering (as specified in H.264/AVC [1]) for coding the low-pass pictures • Avoiding temporal prediction between different groups of pictures (cp. sec. 3.1.3, 3.2.2)
9	**Graceful degradation**	With the proposed coding scheme **graceful degradation under different transmission errors can be provided**. In order to increase the robustness to different transmission errors, and thus the video quality for a transmission over error-prone networks, the above-mentioned features/methods (see *Robustness to different types of transmission errors*) can be employed.
10	Robustness under "best-effort" networks	A certain degree of robustness under "best-effort" networks is inherently provided by the open-loop structure of the proposed coding scheme. One possibility to increase the robustness under "best-effort" networks is to forbid the temporal prediction between the low-pass pictures of different groups of pictures, since thus, each group of pictures can be decoded independently. Furthermore, the concept of redundant slices (as specified in H.264/AVC [1]) can be used to increase the probability that a decoder

		receives the most important VCL NAL units of a bit stream (e.g. low-pass pictures and prediction data arrays of the base layer).
11	10, specially in the presence of server and path diversity	A mechanism that enables improved performance over "best-effort" networks when using server and/or path diversity is not supported.
12	**Colour depth**	Colorimetry information as colour primaries, transfer characteristics, and matrix coefficients can be transmitted as part of the sequence parameter set using the VUI (***Video Usability Information***) syntax as specified in H.264/AVC [1]. Currently, the H.264/AVC [1] standard (and thus the proposed scalable extension) supports the coding of moving pictures in 4:2:0 chroma format with a bit depth of 8 bits per pixel. However, the following features will be supported in the **Professional Extension Amendment** (see [15]) of H.264/AVC, which is currently defined: • The coding of moving pictures with a bit depth of 10 or 12 bits per pixel • The coding of moving pictures in chroma formats with 4:2:2 and 4:4:4 samplings • The coding of moving pictures in RGB format
13	**Coding efficiency performance**	The simulation results presented in sec. 5 indicate that the proposed **coding scheme is capable of providing nearly the same coding efficiency as the state-of-the-art H.264/AVC standard under error-free conditions**. The coding efficiency of an embedded bit stream strongly depends on encoder decisions and/or application requirements as • The selected operating point (Lagrangian parameter used for motion estimation and mode decision) • The GOP size and the decomposition structure • The end-to-end delay • Robustness to transmission errors (e.g. if prediction between low-pass pictures is allowed) • Random access • Memory constraints • The number of scalability levels and the increase in bit rate from one layer to the next In sec. 5, it was especially shown that a single layer version of the proposed coding scheme can provide a coding efficiency superior to that of the state-of-the-art H.264/AVC standard. For three of the four test sequences defined in Test Scenario 2 (see [10]), we could achieve SNR gains of up to 1.5 dB. We could also observe an improvement in subjective quality when coding these sequences in CIF resolution with a frame rate of 30 Hz.
14	Base-layer compatibility	The proposed coding scheme represents a scalable extension of the H.264/AVC standard [1]. Standard compliant H.264/AVC coding can be interpreted as special case of the presented coding scheme. **A base layer can be transmitted as H.264/AVC compliant bit stream**; and actually, we used this feature for generating the submitted bit streams.
15	**Low complexity codecs**	The proposed coding scheme provides **several features that can be exploited for the implementation of low-complexity encoders and/or decoders**. The following features reduce the complexity of both the encoder and the decoder: • Selection of the GOP size. Especially, if groups of a single picture are coded, the complexity reduces to that of H.264/AVC. • Skipping of the update steps in the decomposition/reconstruction process. Due to the open-loop structure of the proposed coding scheme, the update steps at the decoder side can also be skipped if they are performed at the encoder side. • Restriction of the reference index lists. Especially, the reference index lists can be restricted in a way that only uni-directional prediction from

		a single picture is allowed. • Selection of block sizes used for motion-compensated prediction. Especially, an encoder can decide that only 16x16 blocks are used for motion-compensated prediction. • Motion vector accuracy. An encoder can select the motion vectors used for motion-compensated prediction. By selecting only sample-accurate motion vectors, the complex sub-pixel interpolation process can be avoided. However, motion vectors still need to be coded in units of a quarter luma sample. • Selection of slice types that are used for coding the low-pass pictures. Especially, the temporal prediction between the low-pass pictures can be turned off. At the decoder side, it is additionally possible to skip the complex deblocking filter process.
16	**End-to-end delay**	The end-to-end delay of the proposed coding scheme is dependent on • The GOP size • The decomposition structure (the placement of the low-pass picture(s)) • The skipping of the update steps • The number and the location of pictures included in the reference lists used for motion-compensated prediction. **An encoder can adjust these parameters in a way that the end-to-end delay is less than 150 ms**. Actually, it is even possible to enable instantaneous encoding and decoding (cp. [2]) by • Skipping all update steps, and • Using only temporal preceding pictures for motion-compensated prediction. In general, low-delay coding will reduce the coding efficiency of the generated bit streams (cp. sec. 6.5).
17	Random access capability	With the proposed coding scheme, **random access to any or selected low-pass pictures of any or selected groups of pictures** can be provided. In order to enable random access for a low-pass picture, temporal prediction from preceding low-pass pictures (see sec. 3.1.3, 3.2.2) has to be avoided in the base layer and all spatial and quality (SNR) enhancement layers.
18	**Support for coding interlaced material**	For the coding of interlaced material, the **picture- and macroblock-adaptive frame field coding defined in the H.264/AVC standard can be used**. These features can be exploited for both, the decomposition of a group of pictures/frames and the coding of the low- and high-pass pictures/frames. If scalability between interlaced and progressive formats is required, the first decomposition stages shall be performed in a way that either all top or all bottom field pictures are replaced by low-pass pictures and the field pictures with opposite parity are replaced by the corresponding high-pass pictures.
19	**System interface to support quality selection**	In our coding scheme, **all data of the base layer and the enhancement layers are mapped to H.264/AVC NAL units**. Thus, a bit stream can easily be manipulated on packet basis. To extract a bit stream for a specific temporal and spatial resolution as well as a specific quality from an embedded bit stream, only the appropriate packets need to be selected. The NAL units can easily be mapped to popular system layers and transport protocols.
20	**Multiple adaptations**	In the proposed coding scheme, each enhancement layer representation includes the base layer and all subordinate enhancement layers. Thus, **a multiple successive extraction of lower quality bit streams from the initial bit stream is supported**.

6.2 Number of Bits Used for Generating the Provided Sequences

In Table 5, the number of bits and the corresponding bit rates that have been used by the decoder for producing the submitted test sequences are listed.

Table 5: Number of bits and corresponding bit rates used for decoding the submitted test sequences.

Sequence	Resolution	Frame Rate	Target Bit Rate	Number of Bits used for Decoding	Actual Bit Rate used for Decoding
Bus	176 x 144	7.5 Hz	48 kbit/s	235504	47.10 kbit/s
		15 Hz	64 kbit/s	299744	59.95 kbit/s
	352 x 288	15 Hz	128 kbit/s	637592	127.52 kbit/s
			256 kbit/s	1277936	255.59 kbit/s
		30 Hz	512 kbit/s	2555272	511.05 kbit/s
			1024 kbit/s	5098384	1019.68 kbit/s
Foreman	176 x 144	7.5 Hz	48 kbit/s	448392	44.84 kbit/s
		15 Hz	64 kbit/s	618536	61.85 kbit/s
	352 x 288	15 Hz	128 kbit/s	1279592	127.96 kbit/s
			256 kbit/s	2549872	254.99 kbit/s
		30 Hz	512 kbit/s	5116952	511.70 kbit/s
			1024 kbit/s	10238680	1023.87 kbit/s
Football	176 x 144	7.5 Hz	64 kbit/s	540952	62.42 kbit/s
		15 Hz	128 kbit/s	822496	94.90 kbit/s
	352 x 288	15 Hz	256 kbit/s	2218152	255.94 kbit/s
			512 kbit/s	4436984	511.96 kbit/s
		30 Hz	1024 kbit/s	8876528	1024.21 kbit/s
			2048 kbit/s	17753152	2048.44 kbit/s
Mobile	176 x 144	7.5 Hz	64 kbit/s	618056	61.81 kbit/s
		15 Hz	128 kbit/s	743368	74.34 kbit/s
	352 x 288	15 Hz	256 kbit/s	2555888	255.59 kbit/s
			512 kbit/s	5124536	512.45 kbit/s
		30 Hz	1024 kbit/s	10259504	1025.95 kbit/s
			2048 kbit/s	20501304	2050.13 kbit/s

6.3 Software

The software for our proposed coding scheme is written in C++. The provided binaries for the decoder and the bit stream extractor have been compiled on a Windows PC.

The following presented features are not yet implemented in software:

- General decomposition structure with arbitrary bit strings *lowPassPartitioning* and flags *skipUpdate* (see sec. 2.2). Currently, for the parameter *lowPassPartitioning* only the bit strings "0101…" and "1010…" are supported; the flag *skipUpdate* is forced to be equal to 1.
- Coding of interlaced material.

6.4 Random Access

With the proposed coding scheme, random access to any or selected low-pass pictures of any or selected groups of pictures can be provided. In order to enable random access for a low-pass picture, temporal prediction from preceding low-pass pictures (see sec. 3.1.3, 3.2.2) has to be avoided in the base layer and all spatial and quality (SNR) enhancement layers.

The submitted bit streams do not provide random access, since temporal prediction between low-pass pictures is generally used (see sec. 4.1).

In general, when temporal prediction between low-pass pictures of different GOP's is not used and the picture sequence is coded in groups of N pictures and, up to N subband pictures need to be decoded to access a single picture. However, when all update steps are skipped (it is possible to skip the update step

only at the decoder side) and only temporal preceding pictures are used for predicting a picture during decomposition, only up to $n + 1$ (with n being the number of decomposition stages) pictures need to be decoded in order to access a single pictures. The number of pictures that need to be decoded to access a single picture is generally reduced when decoding a representation with reduced temporal resolution. In a spatial or quality enhancement layer, the subband pictures or reconstructed pictures that are used for predicting the subband pictures needed for reconstructing a regarded picture need to be decoded additionally.

6.5 Encoding and Decoding Delay

The end-to-end delay of the proposed coding scheme is dependent on

- The GOP size
- The decomposition structure (the placement of the low-pass picture(s))
- The skipping of the update steps
- The number and the location of pictures included in the reference lists used for motion-compensated prediction.

An encoder can adjust these parameters in order to meet any given delay constraints (up to instantaneous encoding and decoding, cp. sec. 6.1).

The encoding-decoding delay for the submitted bit streams is equal to 1100 ms. At this, a delay of 1033.33 ms is related to the used GOP structure of 32 pictures (see sec. 4.1) for the spatial enhancement layer (CIF resolution), and an additional delay of 66.67 ms is related to the "IBPBP…" structure used for coding the H.264/AVC compliant base layer.

In order to estimate the drop in SNR when a maximum decoding delay of 150 ms is imposed, we have compared the coding efficiency of the submitted bit streams with the coding efficiency of bit streams generated with a simple low-delay version of our encoder. This simple low-delay version of our encoder uses groups of 4 pictures for coding the spatial enhancement layer (CIF) and groups of 2 pictures for coding the spatial base layer (QCIF); for the low-delay version, the base layer is not compliant with H.264/AVC. The bit streams generated in low-delay mode provide the same degree of scalability as the submitted bit streams; the encoding-decoding delay is 100 ms. The rate-distortion performance of both encoder versions is compared in Figure 10–Figure 13. By imposing the maximum encoding-decoding delay of 150 ms, the coding efficiency is decreased by 0 to about 1 dB for the test sequences *Bus*, *Foreman*, and *Football*. For the sequence *Mobile*, a drop in SNR of up to 2 dB has been observed. We believe that the drop in SNR can be reduced if a combination of the above-mentioned parameters is used to design a more efficient low-delay encoding mode.

6.6 Complexity

6.6.1 Motion Compensation

In the proposed coding scheme, the motion-compensated prediction is performed as specified in the H.264/AVC standard [1]. Motion-compensation prediction is generally performed with quarter-sample accuracy for luma samples and eighth-sample accuracy for chroma samples. Prediction values for luma samples at half-sample positions are obtained by applying the one-dimensional 6-tap FIR filter {1, –5, 20, 20, –5, 1}; prediction values for luma samples at quarter-sample positions are generated by averaging samples at integer- and half-sample positions. For chroma samples, the prediction values are always obtained via bi-linear interpolation. For motion-compensated prediction, block size of 16x16, 16x8, 8x16, 8x8, 8x4, 4x8, and 4x4 luma samples can be used. Multi-picture motion-compensated prediction is generally supported. That is, more than one picture can be used as reference for predicting a single picture.

For generating the submitted bit streams, the number of active entries for all reference index lists was always set to 1, and thus, only neighbouring pictures have been used for motion-compensated prediction. Only for the H.264/AVC compliant base layer, 5 reference frames have been used. Thus, a total of 36

frame memories in CIF resolution and 17 frame memories in QCIF resolution are required to decode the highest resolution layer of the submitted bit streams: 32 frame memories in CIF resolution are needed to store and reconstruct a group of pictures (32 pictures, see sec. 4.1) of the enhancement layer (CIF resolution), 4 frame memories in CIF resolution are needed to store the reconstructed low-pass pictures (1 for each CIF layer) that are used for predicting the low-pass picture of the current GOP, and 17 frame memories in QCIF resolution are needed to decode and store the QCIF base layer pictures that are used to predict the low-pass signal of the current GOP as well as the intra macroblocks in the high-pass signals of the current GOP.

6.6.2 Spatial Transform

For the coding of prediction error signals (intra prediction or motion-compensated prediction), transform coding as specified in H.264/AVC [1] is applied. The transformation is applied to blocks of 4x4 samples. In INTRA_16x16 mode, an additional 4x4 transform is applied to the 4x4 DC coefficients of the luma component of a macroblock; and similarly, an additional 2x2 transform is applied to the four DC coefficients of each chroma component of a macroblock. The transformation can be realized using only additions and bit-shifting operations of 16-bit integer values.

6.6.3 Decoding Complexity

We estimate that the decoding complexity for the proposed coding scheme is increased by a factor of 2–3 in comparison to standard H.264/AVC coding:

- Assuming that not all frames are coded using B slices with an H.264/AVC compliant coder, the complexity of the motion-compensation process is increased by a factor of 2–3 due to the additional motion compensation process in the update steps.
- The complexity of the deblocking filter process is similar to that of H.264/AVC, since in addition to the low-pass pictures, the deblocking filter is only applied to pictures that are generated in the prediction steps at the decoder side.
- The complexity of the entropy decoding is similar to that of H.264/AVC, since for a given entropy-coding algorithm, the complexity is mainly determined by the bit rate of the input bit stream.
- For spatial enhancement layers, the decoding complexity is further increased, since additionally all pictures of the spatial base layer (with reduced spatial resolution) that are used for predicting the low-pass pictures and the intra-macroblocks in the high-pass pictures of the spatial enhancement layer need to be reconstructed.

For decoding a representation with reduced temporal resolution, the decoding complexity is accordingly reduced, since only a subset of the motion-compensated prediction and update steps, the deblocking operations, and the entropy decoding is performed. Similarly, the decoding complexity is reduced when decoding a representation with reduced spatial resolution. For decoding a representation with reduced bit rate (quality), only the complexity of the entropy decoding is accordingly reduced.

It should be noted that due to the open-loop structure of the proposed coding scheme low-complexity decoders could skip the update steps as well as the deblocking filter operations while still reconstructing a picture sequence of reasonable quality.

6.7 Rate Control

In our current encoder implementation, the bit-allocation for each group of pictures and each scalability layer is controlled by a single parameter QP_{r0} as described in sec. 4.4. Thus, the spatio-temporal window used for rate control is one group of pictures (for the submitted bit streams, a group of pictures contains 32 pictures of the original 30 Hz sequence). However, with the implemented algorithm, the number of bits used for each GOP and scalability layer is not forced to be constant. A several degree of bit-rate variations between successive GOP's is allowed; the main objective is the average bit rate of the produced base and enhancement layer bit streams.

6.8 Granularity of Scalability Levels

The spatial and/or temporal resolution from one layer to the next can be increased by any factor k / m with k > m > 1 (see sec. 3.2.1, 3.2.3). For the quality (SNR) scalability, the granularity can arbitrarily be adjusted (see sec. 3.2.2).

6.9 Usage of a Base Layer

Our coding scheme generally uses a base layer. The base layer can be coded in a way that it is compliant with the H.264/AVC standard; however, it is not required to be H.264/AVC compliant. For the submitted bit streams, we actually used an H.264/AVC compliant base layer.

6.10 Compliance with Existing Standards

The H.264/AVC standard could be interpreted as specialization of the proposed coding scheme. When only a base layer (in terms of spatial resolution and quality (SNR)) is coded using groups of a single picture, the produced bit stream is compliant with the H.264/AVC standard [1].

7 Summary

We presented a scalable extension of H.264/AVC that requires only a few adjustments for enabling temporal, spatial, and quality scalability within a block-based motion-compensated temporal lifting framework. This proposed open-loop approach of motion compensation includes multiple reference pictures as well as adaptive choice between two lifting representations according to uni-directional prediction and bi-prediction on a block bases. Furthermore, the prediction and update steps can be switched off on a block basis, and thus intra coding is enabled for blocks that cannot be reasonably represented using motion-compensated prediction. As a distinctive feature, motion parameters for the update process are derived from motion parameters estimated for the corresponding prediction steps in a way that they can still be represented by the H.264/AVC syntax. Experimental results indicate that the proposed scalable extension of H.264/AVC is capable of providing a coding efficiency nearly comparable to that of an original H.264/AVC compliant encoder; the coding efficiency strongly depends on the application requirements. A single layer version of the proposed coding scheme can even provide a coding efficiency superior to that of the state-of-the-art H.264/AVC standard.

References

[1] ITU-T and ISO/IEC JTC1, “Advanced Video Coding for Generic Audiovisual Services,” ITU-T Recommendation H.264 – ISO/IEC 14496-10 AVC, 2003.

[2] J.-R. Ohm, “Complexity and delay analysis of MCTF interframe wavelet structures,” ISO/IEC JTC1/WG11 Doc. M8520, July 2002.

[3] D. Taubman, “Successive refinement of video: fundamental issues, past efforts and new directions,” *Proc. of SPIE (VCIP 2003)*, vol. 5120, pp. 649-663, July 2003.

[4] M- Flierl and B. Girod, “Video coding with motion-compensated lifted wavelet transforms,” *Proc. of PCS*, pp. 59-62, April 2003.

[5] M. Flierl, “Video Coding with Lifted Wavelet Transforms and Frame-Adaptive Motion Compensation,” *Proc. of VLBL*, pp. 243-251, Sep. 2003.

[6] W. Sweldens, “A custom-design construction of biorthogonal wavelets,” J. Appl. Comp. Harm. Anal., vol. 3 (no. 2), pp. 186-200, 1996.

[7] ITU-T and ISO/IEC JTC1, “Generic Coding of Moving Pictures and Associated Audio Information – Part 2: Video,” ITU-T Recommendation H.262 – ISO/IEC 13818-2 (MPEG-2), 1994.

[8] ITU-T, “Video Coding for Low Bitrate Communication, “ ITU-T Recommendation H.263, Version 1: Nov. 1995, Version 2: Jan. 1998.

[9] ISO/IEC JTC1, “Coding of audio-visual objects – Part 2: Visual, “ ISO/IEC 14496-2 (MPEG-4 Visual), Version 1: April 1999, Amendment 1 (Version 2), Feb. 2000.

[10] ISO/IEC JTC1, "Call for Proposals on Scalable Video Coding Technology," ISO/IEC JTC1/WG11 Doc. N5958, Oct. 2003.

[11] T. Wiegand, et al, "Rate-Constrained Coder Control and Comparison of Video Coding Standards," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, July 2003.

[12] T. Wiegand (ed.), "Working draft number 2, revision 8 (WD-2 rev 8)," Joint Video Team of ISO/MPEG and ITU-T VCEG, Doc. JVT-B118r8, Apr. 2002.

[13] ISO/IEC JTC1, "Requirements and Applications for Scalable Video Coding," ISO/IEC JTC1/WG11 Doc. N6025, Oct. 2003.

[14] T. Rusert and C.-J. Tsai, "Generation of original Test Sequences and AVC Anchors for Call for Proposals on Scalable Video Coding Technology," ISO/IEC JTC1/WG11 Doc. N10402, Dec. 2003.

[15] T, McMahon, et al, "Draft Prof. Ext. Amendment," Joint Video Team of ISO/MPEG and ITU-T VCEG, Doc. JVT-H037r1, May 2003.