Motion-Refined Rewriting of H.264/AVC-Coded Video to SVC Streams

Jan De Cock, Stijn Notebaert, Peter Lambert, and Rik Van de Walle

Ghent University – IBBT
Department of Electronics and Information Systems – Multimedia Lab
Gaston Crommenlaan 8 b 201
B-9050 Ledeberg-Ghent, Belgium

Abstract

In this paper, we discuss motion-refined rewriting of single-layer H.264/AVC streams to SVC streams with multiple quality layers. First, we elaborate on techniques we developed for efficient rewriting of residual data from H.264/AVC to SVC. We investigate if rate-distortion performance can further be improved by extending these architectures with motion refinement techniques, which exploit the inter-layer motion prediction mechanisms available in SVC. For optimum performance, we discuss a fast rate-distortion technique based on Lagrangian relaxation. Although motion refinement in the transform-domain leads to extra distortion in the bitstream, we show that our rate-distortion model successfully takes into account both base and enhancement layer rate and distortion during optimization. Implementation results show that motion-refined rewriting in the transform domain can increase rate-distortion performance, with gains of up to 0.5 dB for the SVC base layer. The presented rewriting architectures significantly reduce the computational complexity when compared to reencoding, with a speed-up by a factor of forty or more, even in the case of motion refinement.

Key words: Transcoding, rewriting, H.264/AVC, SVC.

1. Introduction

Recently, joint efforts of MPEG and VCEG have led to the standardization of a new state-of-the-art scalable video codec [1]. This scalable extension of H.264/AVC, usually referred to as SVC, makes it possible to encode scalable video bitstreams containing several quality, spatial, and temporal layers.

By parsing and extracting, lower layers can easily be obtained, hence providing different types of scalability in a flexible way.

In order to benefit from these adaptation features, scalability has to be provided during encoding. This implies that existing H.264/AVC content cannot benefit from the scalability tools in SVC due to the lack of intrinsic scalability provided in the bitstream at encoding time. Nonetheless, it would be beneficial for broadcasters and content distributors to have scalable bitstreams at their disposal in order to allow easy adaptation of the video streams to a wide range of devices, or to tailor the bit rate to the available network bandwidth. To achieve conversion from single-layer streams to scalable streams, efficient techniques for migration of existing content to a scalable format are desirable.

As a straightforward solution, decoding and re-encoding could be used. This, however, requires a tremendous amount of time for conversion, given the highly complex SVC encoding process. Among others, the motion estimation process that has to be repeated in every layer results in severely increased computational complexity.

As an alternative technique, transcoding can be used. Transcoding is a popular technique for adaptation of video content that does not impose constraints on the original bitstream, i.e., the bitstream does not have to be scalable to allow transcoding [2]. One of the main goals in designing transcoding architectures is to obtain an architecture that performs adaptation at a computational cost significantly lower than decoding and re-encoding, while achieving rate-distortion performance close to that of the cascaded decoder-encoder. The speed-up in execution time can be achieved by reusing information from the incoming bitstream in an intelligent way. In the past, different transcoding solutions have been presented, with architectures that provided features such as quality, spatial, and temporal scalability [2, 3, 4].

Due to its computational efficiency, transcoding can be used for introducing scalability in compressed, single-layer bitstreams. In this way, re-encoding can be avoided when migrating legacy content to a scalable format. A number of techniques have been proposed in the past for introducing scalability in compressed bitstreams. The technique of introducing multiple layers starting from a single-layer bitstream was studied in the context of data partitioning in [5, 6]. Here, for MPEG-2, a data partitioning scheme was proposed based on the determination of the (causally) optimal breakpoint in DCT coefficient blocks. In [7], the problem of transcoding MPEG(-2/4) streams to (MPEG-4) Fine-Grain Scalability (FGS) was studied. A closed-loop solution, based on a simplified cascade of decoder and encoder, was found to be useful in the context of elastic storage of compressed video content. In [8], a technique was studied for transcoding of hierarchically coded H.264/AVC streams to streams with multiple FGS quality layers. Hierarchically coded B-pictures were used to achieve combined temporal and quality scalability, and to obtain improved rate-distortion performance. The benefit of using hierarchical B pictures on R-D performance was shown in [9]. Based on spatial and temporal drift compensation techniques in [10], we introduced a technique for H.264/AVC-to-SVC transcoding to two quality layers in [11]. A reduced-complexity technique based on bitstream rewriting [12] was proposed in [13], which was shown to be able to outperform the technique based on drift compensation.

Most existing techniques only focus on residual data transcoding, i.e., without taking into account the motion data in the bitstream. In these schemes, residual data is distributed (or refined) among the different layers, but all motion data is concentrated in the base layer. As it turns out, however, for larger reductions of the base layer bit rate it is beneficial to also adjust the motion parameters, i.e., motion vectors, macroblock partitioning, reference picture indices, etc. By doing this, coarser motion information is included in the base layer while enhancement layers contain further refinements of the motion data.

For the complexity reasons mentioned above we avoid decoding (which requires pixel-domain reconstruction) and re-encoding (with its included motion estimation process). Instead, we start from techniques we developed for residual data rewriting and extend these to include motion vector refinement. As a result, our architecture operates completely in the transform domain and avoids the time-consuming steps involved in decoding and re-encoding. Adjustment of the motion parameters should be performed in a prudent way, since changed values could lead to misprediction during motion compensation. This could lead to significant distortion and artifacts which could propagate and cause drift in the video stream. Because of these reasons it is important to be able to reliably estimate the distortion introduced by changing motion parameters, and to provide a reliable model for rate-distortion trade-off in order to improve overall R-D performance.

Although working in the transform domain somehow limits the freedom of adjustment of motion parameters (large adjustments would incur significant errors), we show that our model for motion refinement can lead to a further reduction of the bit rate without causing distortion that would result in a negative net rate-distortion result.

Further, we provide a multi-layer control model that allows to trade off base layer vs. enhancement layer R-D performance. Hence, we provide liberty for our implementation to distribute motion data bits among the most appropriate layer. Multi-layer *encoder* control is currently not included in the JSVM reference encoder software (instead, a bottom-up process is followed). An initial assessment of its beneficial impact on rate-distortion efficiency, however, has been studied in [14]. Here, we apply the concepts to the rewriting case. Although motion-refined encoding was not studied in [14], in our case we extend the concepts to include the more general case of motion refinement.

The remainder of this paper is organized as follows. In Sect. 2, more information is given on residual data rewriting. In Sect. 3, we provide details on motion vector prediction in H.264/AVC and SVC. Our techniques for motion data refinement are described in Sect. 4. The multi-layer control model for H.264/AVC-to-SVC rewriting is introduced in Sect. 5. In Sect. 6, we give implementation results for the proposed techniques. We conclude in Sect. 7.

2. Residual data rewriting from H.264/AVC to SVC

2.1. Inter-layer residual prediction and inter-layer intra prediction in SVC

First, we briefly introduce the techniques for inter-layer residual data prediction in SVC. A distinction is made between inter-layer *residual* prediction and inter-layer *intra* prediction.

2.1.1. Inter-layer residual prediction

In SVC, inter-layer residual prediction allows efficient compression of residual coefficients. By using inter-layer residual prediction, residual coefficients of enhancement layers are represented efficiently by only coding the differences between the ‘fine’ coefficients in the enhancement layer(s) and the ‘coarse’ coefficients found in the lower reference layer. During H.264/AVC-to-SVC transcoding, both the base layer ‘coarse’ coefficient and the enhancement layer refinement values need to be derived from the original ‘fine’ coefficients as found in the incoming single-layer bitstream. Preferably, this should be done in a way that avoids information loss, i.e., all residual information found in the original bitstream should be reallocated in the different layers of the SVC stream. Otherwise stated, when decoding the SVC stream (in case all layers are present) the obtained coefficients should be identical to the original coefficients in the single-layer H.264/AVC stream.

2.1.2. Inter-layer intra prediction

If an enhancement layer macroblock is coded using *base mode flag* equal to 1, and the co-located macroblock in the reference layer is intra-coded, the prediction is formed by using inter-layer *intra* prediction. This macroblock type (also known as an I_BL macroblock) can be used not only in intra-coded pictures (I pictures), but also in predictive (P) or bidirectionally predictive (B) pictures, as an addition to the 'traditional' H.264/AVC Intra_$N \times N$ macroblock types (where N equals 4, 8, or 16). In the SVC design, a restriction was provided that only allows inter-layer intra prediction when the intra macroblock in the lower layer uses *constrained* intra prediction. Constrained intra prediction indicates that intra prediction in these macroblocks can only be based on other intra-coded macroblocks (and not on inter-coded macroblocks).

2.2. Open-loop H.264/AVC-to-SVC transcoding

During H.264/AVC-to-SVC conversion, the residual coefficients as found in the single-layer bitstream are requantized and split among the different layers. In the output SVC stream, the base layer contains the residual coefficients, requantized using the highest quantization parameter (QP), while the enhancement layers, with lower QPs, contain refinement residual data.

This is illustrated in Fig. 1 for two layers (one base layer and one enhancement layer). Here, the original coefficient R_I is dequantized using the original QP_I and subsequently requantized using the base layer QP_B, leading to coefficient R_B. In order to obtain the enhancement layer coefficient R_E, the difference is calculated between the dequantized versions of the original coefficient R_I and the base layer coefficient R_B. This difference is requantized using the enhancement layer QP_E.

Two problems arise for this open-loop transcoding architecture:

- There will be cases in which the distribution of the coefficients over the different SVC layers will not lead to identical reconstruction when compared to the original bitstream. In other words, loss of information will occur after transcoding. This loss is due to the H.264/AVC (and SVC) quantizer design and is preferably avoided, since it can lead to drift in the output SVC bitstream.

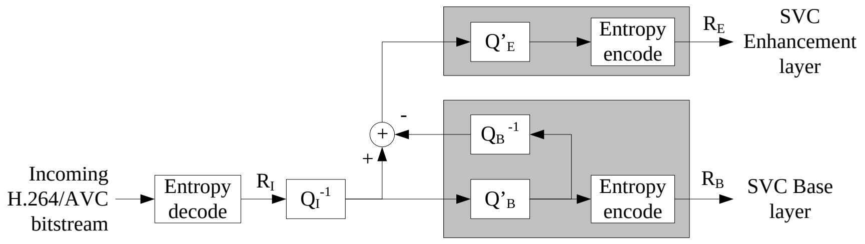


Figure 1: Open-loop transcoding from H.264/AVC to SVC.

- The architecture in Fig. 1 does not take into account intra-coded macroblocks. When working in the transform domain (i.e., without performing a full pixel-domain reconstruction), intra-coded macroblocks cannot benefit from inter-layer intra prediction since typically *constrained* intra prediction is not used in the original H.264/AVC bitstreams.

2.3. Bitstream rewriting from SVC to H.264/AVC

Bitstream rewriting was added to the SVC design to allow low-complexity combination of coarse-grain quality scalability (CGS) dependency layers or medium-grain quality scalability (MGS) quality layers to a single-layer H.264/-AVC stream, hereby providing backward compatibility for existing H.264/-AVC decoding equipment.

The technique for rewriting an SVC stream with multiple dependency layers to a single-layer H.264/AVC stream was introduced in [12]. At the time this was applied to CGS (the MGS functionality was added later on in the standardization process), but since the difference between CGS and MGS is primarily limited to a number of high-level syntax changes, the technique is also applicable to MGS quality levels.

A number of changes were required to the SVC syntax and coding tools in order to allow this kind of rewriting functionality. In particular, two major aspects of the decoding process were modified [15].

- For inter-layer intra prediction the prediction signal is no longer formed by the reconstructed intra signal of the reference layer, but spatial intra prediction as in single-layer H.264/AVC is performed in the target layer. The residual signal is formed in the same way as for motion-compensated prediction (MCP) macroblock types (see Fig. 2). This also means that the constrained intra prediction requirement for the lower layers is dropped for I_BL macroblocks, in the case that the bitstream rewriting functionality is applied.

- Residual prediction is performed in the transform coefficient level domain. Not the scaled transform coefficients, but the quantization levels are scaled and accumulated during decoding.

In Fig. 2, the adapted reconstruction process of intra-coded and MCP macroblocks is shown. In the SVC specification, the inverse and forward quantization steps Q_B^{-1} and Q'_E are combined, resulting in efficient scaling formulas.

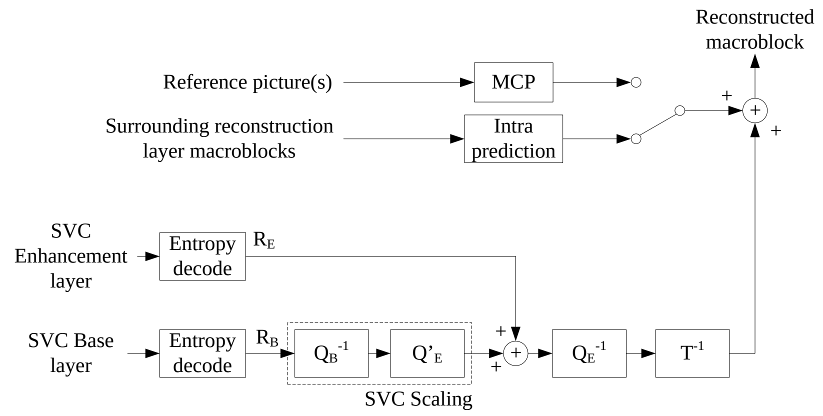


Figure 2: Reconstruction of intra-coded and MCP macroblocks using adapted coefficient accumulation process.

2.4. Open-loop rewriting from H.264/AVC to SVC

By exploiting the bitstream rewriting functionality in H.264/AVC-to-SVC transcoding, the two problems mentioned in Sect. 2.2 can be solved. The open-loop architecture is changed in two ways, resulting in the design shown in Fig. 3.

- Using the rewriting functionality has the advantage that perfect reconstruction is achieved in the top layer of the SVC stream. The rewriting functionality allows for identical accumulation and reconstruction of the coefficients.

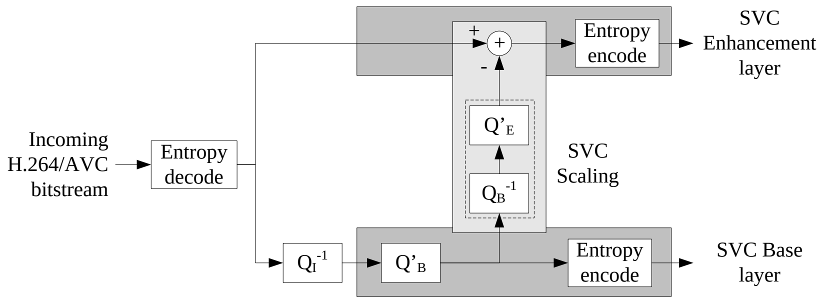


Figure 3: Open-loop rewriting architecture for intra-coded and MCP macroblocks based on changed coefficient accumulation process.

- As mentioned, the constrained intra prediction requirement is canceled when bitstream rewriting is enabled. This has the benefit that the open-loop transcoding architecture (Fig. 3) can also be applied to intra-coded pictures and intra-coded macroblocks in P and B pictures. Similar to temporal drift, however, spatial drift will arise which is caused by spatial dependencies. Due to the larger number of dependent blocks [16], spatial drift will propagate rapidly, and result in serious artifacts in video streams extracted from lower layers.

Slight changes are made to the open-loop transcoding architecture, resulting in the rewriting architecture shown in Fig. 3. The standardized SVC scaling process can be used for transform coefficient accumulation. It can be seen that the coefficient subtraction process in this architecture corresponds to the coefficient accumulation process in Fig. 2 for MCP blocks.

2.5. Spatial drift and advanced bitstream rewriting from H.264/AVC to SVC

By exploiting the bitstream rewriting functionality, information loss is avoided during the H.264/AVC-to-SVC conversion for the top layer, i.e., perfect reconstruction is achieved when all layers are present in the SVC bitstream. This, however, does not apply to the lower layers. Due to the absence of a full (pixel-domain) reconstruction loop requantization errors will propagate in these lower layers, both spatially and temporally due to intra prediction and MCP, respectively. In particular, drift due to intra prediction has been shown to be a major issue in H.264/AVC video coding [17]. To overcome the intra drift issue, a number of strategies can be used, which will be discussed in the remainder of this section. In particular, the goal is to keep the computational complexity as low as possible when compared to open-loop rewriting of the intra-coded macroblocks (Fig. 3). In any case, quality should not be compromised.

2.5.1. Intra copy for intra-coded macroblocks (IC)

A low-complexity technique to avoid spatial drift is to concentrate all intra-coded data in the SVC *base* layer, while the enhancement layers contain no residual data (the *base mode flag* and I_BL macroblock type will allow the base layer data to propagate to higher layers). This can be regarded as a low-complexity technique, since only high-level syntax changes are required. In the rewriter, the incoming residual data can simply be copied to the output bitstream. The rate-distortion performance of the output SVC sequence will benefit from this approach, since layered coding results in a rate-distortion penalty when compared to single-layer coding. Obviously, this approach offers no flexibility regarding rate distribution whatsoever for these macroblocks. In practice, concentrating the residual data in the base layer will be implemented by only increasing the macroblock QP for non-intra coded macroblocks in the base layer, while the QP will be unchanged for intra macroblocks. This is reflected by a non-zero value of the *mb_qp_delta* syntax element in the bitstream.

2.5.2. Rewriting with spatial compensation (RW)

In order to alleviate the spatial drift problem of open-loop transcoding, another architecture can be introduced for H.264/AVC-to-SVC transcoding which provides full flexibility regarding rate redistribution, and is based on the changes introduced by bitstream rewriting. Similar to spatial drift compensation techniques in single-layer transrating [10], it is possible to perform transform-domain operations on intra-coded macroblocks in P and B pictures, hereby avoiding reconstruction of surrounding inter-coded macroblocks. Additionally, since constrained intra prediction is no longer a requirement in lower layers, this architecture benefits from inter-layer residual prediction, as is shown in Fig. 4. When compared to the open-loop transcoding architecture, a single spatial compensation loop is introduced which neutralizes the loss of information during requantization. In this way, the loss of information will not propagate due to intra prediction. This architecture has the advantage that no decoding is necessary for MCP blocks, yet full flexibility is possible for requantization of intra-coded macroblocks in P and B pictures. This provides an advantage over the intra copy or intra-layer

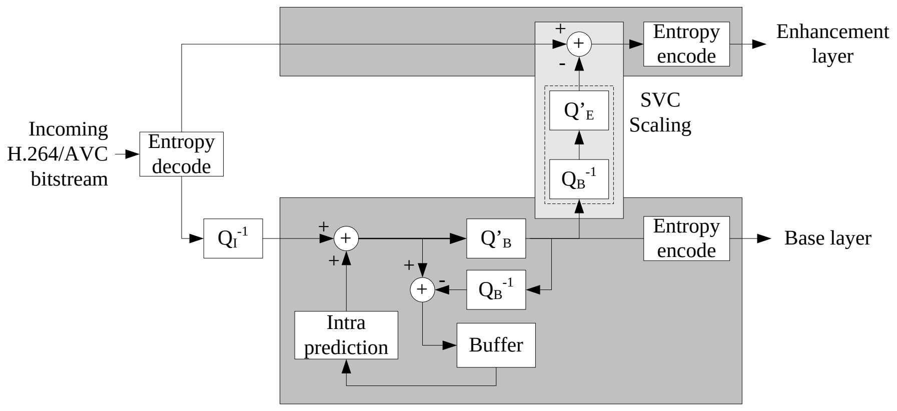


Figure 4: Single-loop transcoding architecture with spatial compensation for non-constrained intra-coded macroblocks.

intra prediction architectures, which lack either rate distribution flexibility or rate-distortion performance.

Although spatial error propagation has a decisive impact on the total amount of drift in the transcoded sequences, it might be beneficial to add temporal compensation for inter-coded macroblocks to further improve quality of the lower SVC layers (note that no drift is present in the top layer due to perfect accumulation). Although temporal compensation improves quality, complexity will significantly increase due to the added MCP loops.

3. Motion vector prediction

In the previously discussed rewriting architectures, we only examined residual data rewriting. For these approaches, motion data was copied from the input H.264/AVC stream to the output SVC stream. In the SVC stream, the motion data is concentrated in the base layer, and is identical for all layers. In the following sections, we investigate if refining (or optimizing) the motion data can be used to further improve rate-distortion efficiency in the desired layer(s).

We benefit from the motion prediction mechanisms in the SVC design to allow small changes in motion information between the different layers to be coded efficiently in the bitstream. These mechanisms are explained briefly in the remainder of this section.

3.1. Motion vector prediction in H.264/AVC

In order to obtain accurate prediction, H.264/AVC allows further partitioning of macroblocks in 16×8, 8×16, and 8×8 macroblock partitions. 8×8 blocks can further be split into two 8×4, 4×8, or four 4×4 submacroblock partitions. Each (sub)macroblock partition has an associated motion vector, referring to its best match in the reference picture. In the case of B pictures, bidirectional prediction can be used. Here, a (weighted) prediction signal is formed by combining a forward and a backward prediction block, as indicated by their associated motion vector. Motion vectors are coded efficiently by using median motion vector prediction. For each (sub)macroblock partition, the motion vector is obtained by combining a motion vector predictor (mvp_i) with the motion vector difference (mvd_i), i.e., $\mathrm{mv}_i = \mathrm{mvp}_i + \mathrm{mvd}_i$. The motion vector predictor is derived based on the motion vector information of neighboring macroblock partitions. In H.264/AVC, the prediction is formed by taking the median of the motion vectors of three surrounding macroblock partitions (if available). The values for mvd_i are coded in the bitstream and represent a significant part of the total bit rate.

Regions of uniform motion will lead to accurate motion vector prediction, resulting in small motion vector differences. In H.264/AVC, (sub)macroblocks that contain no motion vector differences (i.e., the case where $\mathrm{mv}_i = \mathrm{mvp}_i$) are represented efficiently by using Skip or Direct modes. In P slices, the P_Skip mode can be used. For this mode, neither motion vector difference nor residual data is coded for the entire macroblock. In B slices, apart from the B_Skip mode, B_Direct prediction can be used for 16×16 macroblocks or 8×8 macroblock partitions. For B_Direct macroblocks or macroblock partitions, residual data can still be coded in the bitstream. For both modes, a choice can be made between spatial and temporal motion vector prediction.

3.2. Motion vector prediction in SVC

Since the coding characteristics (such as quality) vary among the SVC layers, it is beneficial to reflect this difference by tuning the motion parameters in every layer. The possibility to update and refine motion information in successive layers was provided in the SVC design. This allows different motion vectors or reference indices to be used for the same macroblock in different layers.

Two separate techniques can be used in SVC to exploit motion information redundancy in the bitstream.

3.2.1. Intra-layer motion prediction

Similar to single-layer H.264/AVC, median motion vector prediction can be used in every layer. The motion vectors of surrounding (sub)macroblock partitions in the same layer are used to form the motion vector prediction.

An important difference, however, occurs for Direct macroblocks in B slices (i.e., B_Direct_16x16 macroblocks, or B_8x8 macroblocks containing B_Direct_8x8 submacroblocks). In enhancement layers, the derivation for motion vectors in Direct macroblocks is changed when compared to the H.264/AVC Direct mode. This has to be taken into account when a Direct mode is used in the enhancement layer. Under certain conditions, this leads to a recalculation of Direct modes.

3.2.2. Inter-layer motion prediction

Although the motion information will not be identical, a lot of similarity will still be found between the different layers. In many cases, this similarity will lead to a more efficient prediction of the motion information. In SVC, inter-layer redundancy between motion information in base and enhancement layers can be exploited by taking the reference layer motion vector as prediction of the current motion vector. As a result, only the difference between both motion vectors needs to be sent in the current layer.

The choice of intra-layer or inter-layer motion vector prediction is indicated by using the *base mode flag* and *motion prediction flags*. When the *base mode flag* is set, all partitioning information, reference indices, and motion vectors are copied from the reference layer to the reconstruction layer, and are reused as such for motion-compensated prediction of the macroblock. If the *base mode flag* is not set, for every macroblock partition, a *motion prediction flag* can be set, which indicates whether a motion vector prediction needs to be formed from the reference layer motion information (i.e., inter-layer motion prediction), or via 'traditional' median prediction (i.e., intra-layer motion prediction).

4. Motion data rewriting

While the original motion information is optimized for the bit rate of the incoming bitstream (or of the top layer of the outgoing SVC stream), this is not necessarily the case for the lower layers of the output SVC stream. When the quality gap between successive layers becomes larger, it is likely that rate-distortion efficiency in the lower layers will benefit from a change in motion parameters. To accomplish this, we examine the potential rate-distortion gain of tweaking motion information for these lower layers. Since a change in motion information induces a change in the motion-compensated prediction signal, a careful examination needs to be made of the change in both the rate and distortion. Similar to the approach used for redistribution of residual data, we avoid loss of motion information. This means that for the top layer, motion information will be identical to that of the incoming bitstream. This also implies that a reduction of motion data in lower layers will have to be compensated by refinement of the data in the higher layers. The cost of these refinement bits is dictated by the accuracy of the SVC motion prediction mechanisms.

First, we lay out our approach for motion refinement in lower layers. Next, we discuss the rate calculation and distortion estimation techniques that will be used for rate-distortion optimized motion decision in Sect. 5. For clarity, we discuss the techniques for two quality layers.

4.1. Motion refinement

As mentioned in Sect. 3, H.264/AVC allows a large degree of flexibility in macroblock partitioning, with (sub)macroblock partitions down to 4×4 pixels. In lower-rate bitstreams, larger block sizes become more dominant, and the amount of submacroblock partitions tends to decrease. Hence, the most natural way of refining mode decisions for lower bit rates is by merging partitions, if the distortion introduced by the merging operation is small enough. During H.264/AVC-to-SVC conversion, the rate-distortion cost of merging macroblock partitions needs to be examined. More information on the derivation of rate and distortion is given in Sect. 4.2 and 4.3.

We examine in successive steps if macroblock partitions can be merged together. If two merged (sub)macroblock partitions use the same motion vector and reference index, no loss is incurred during the merging operation. If the merged macroblock partitions contain different motion vectors (which is typically the case), however, a mismatch arises and the introduced distortion needs to be estimated.

To derive the most appropriate output motion vector for the merged partition we successively evaluate the cost for each of the input motion vectors of the constituting partitions. This is illustrated in Fig. 5 for the case of submacroblock partitions. If the initial macroblock partition consisted of four 4×4 submacroblock partitions and the case of 8×4 partitions is examined, the two constituting motion vectors are evaluated for each partition, i.e.,

MV_0 vs. MV_1, and MV_2 vs. MV_3. In a similar way, the cost of 4×8 partitions is examined by evaluating MV_0 vs. MV_2, and MV_1 vs. MV_3. All four motion vectors are evaluated to obtain the cost of an 8×8 partition. When merging *submacroblock* partitions, reference indices and prediction directions (forward, backward, or bidirectional prediction) do not have to be taken into consideration, since these are identical for all submacroblock partitions of a single 8×8 partition.

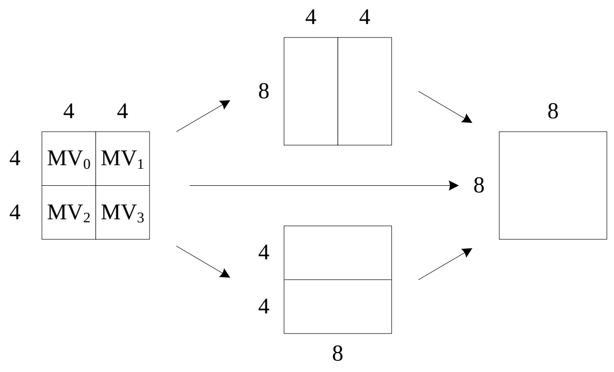


Figure 5: Submacroblock partition merging.

When merging macroblock partitions (8×8 and larger), however, special care has to be taken to avoid merging partitions that contain motion vectors pointing to different reference pictures. Reference picture indices can have a granularity down to 8×8 pixels (i.e., all submacroblock partitions within a single 8×8 block will refer to the same reference picture). If macroblock partitions with different reference indices would be merged, serious artifacts would arise in the decoded video stream, in particular when the temporal distance between the two reference pictures increases.

This problem is aggravated in B pictures, where different prediction directions can be used for each macroblock partition, i.e., reference pictures can be selected from different lists (forward prediction list, backward prediction list, or both). When bidirectional prediction is used, the partition is predicted based on a weighted sum of prediction signals.

Since merging partitions with different reference indices or prediction directions would cause artifacts in the transcoded bitstream, we avoid this situation, and only consider merging partitions with identical reference indices and prediction direction. This is illustrated in Fig. 6 for an example where two bidirectionally predicted 8×8 partitions and two forward predicted

8×8 partitions are merged to a macroblock with 8×16 partitioning.

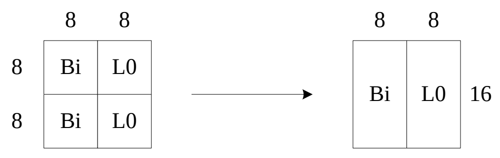


Figure 6: Macroblock partition merging.

After each possible merge operation, the rate-distortion cost is determined, as will be explained in Sect. 5. The rate and distortion are determined as described in the following subsections.

4.2. Rate calculation

Different motion-related syntax elements in base and enhancement layer syntax contribute to the output motion data rate. For the base layer, the macroblock type and if necessary submacroblock types, reference picture indices, and motion vector differences need to be transmitted. If the macroblock is skipped, only a *macroblock skip run* (CAVLC entropy coding) or *macroblock skip flag* (CABAC) needs to be sent (one bit or less per skipped macroblock).

For the enhancement layer, a number of scenarios are possible, depending on the choice for inter-layer or intra-layer motion prediction. In case all motion information of a macroblock can be reused from the base layer, only the *base mode flag* is set and coded in the bitstream. If this is not the case, but a reliable approximation can be formed based on the base layer motion information, *motion prediction flags* can still be used to indicate that the reference indices can be copied from the base layer, and that a predictor can be formed based on the base layer. As an alternative, intra-layer motion vector prediction can be used to achieve the same result, and might result in improved coding efficiency in certain cases.

A trade-off needs to be sought between base layer rate and enhancement layer rate. If we avoid loss of information in the H.264/AVC-to-SVC conversion, a reduction of information in the base layer will have to be counterbalanced by inserting refinement information in the enhancement layer. More information on this trade-off is given in Sect. 5.

4.3. Distortion estimation

As shown in [18, 19], the distortion (D) introduced by motion vector variation can be estimated in the transform domain based on the picture power spectrum. The distortion, expressed as the SSD between the prediction signal p_i of the input motion vector and the prediction signal p_o of the refined output motion vector, i.e., $D = \sum_{m=1}^{4}\sum_{n=1}^{4}(p_i[m,n] - p_o[m,n])^2$ can hence be approximated as follows:

$$\begin{aligned} D &\approx \frac{1}{(2\pi)^2}\iint_{]-\pi,\pi]} S_i(\omega_1,\omega_2)\cdot(\boldsymbol{\omega}\Delta mv)^2\, d\boldsymbol{\omega} \\ &\approx \phi_x \Delta mv_x^2 + \phi_y \Delta mv_y^2 \end{aligned}$$

with $S_i(\omega_1,\omega_2)$ being the power spectral density of the prediction signal associated with the input motion vector mv_i. $\Delta mv = (\Delta mv_x, \Delta mv_y)$ expresses the difference between the input and candidate output motion vectors mv_i and mv_o.

ϕ_x and ϕ_y are determined as follows:

$$\phi_x = \frac{1}{(2\pi)^2}\iint_{]-\pi,\pi]} S_i(\omega_1,\omega_2)\cdot\omega_1^2\, d\omega_1 d\omega_2$$

and

$$\phi_y = \frac{1}{(2\pi)^2}\iint_{]-\pi,\pi]} S_i(\omega_1,\omega_2)\cdot\omega_2^2\, d\omega_1 d\omega_2.$$

The power spectrum can be obtained by approximating the FFT using the 4×4 integer transform in H.264/AVC. Note that since H.264/AVC uses a 4×4 integer transform, a normalization is required to obtain 'traditional' DCT coefficients.

The above integrals for ϕ_x and ϕ_y are discretized by setting the frequencies ω_1 and ω_2 to $\pm\frac{k\pi}{N}$ with $k = 0,\ldots,3$, and $N = 4$.

5. Multi-layer control for H.264/AVC-to-SVC rewriting

The rate and distortion caused by every motion refinement step are obtained using the techniques discussed in Sect. 4. As mentioned, a reduction of the rate in a lower layer will lead to an increase of the bit rate in higher layers, leading to a trade-off between the different layers. The decision whether or not the evaluated refinement will be executed will depend on the impact of the rate and distortion in every layer. We use a multi-layer control mechanism which attaches a weight factor to every layer. The value of this weight factor depends on the scenario in which the rewriter is used. Based on the weight factors and the rate and distortion costs in every layer, we end up with a joint optimization approach. This multi-layer control mechanism is discussed in the remainder of this section.

We discuss the case for two layers, i.e., the base layer (indicated as *layer 0*) and one enhancement layer (*layer 1*). In this case, *base* layer coding decisions are made by minimizing

$$D_0(\boldsymbol{p_0}) + \lambda_0 R_0(\boldsymbol{p_0}),$$

where $\boldsymbol{p_i}$ encompasses the mode decisions m_i and motion vectors v_i for each layer i, respectively. This leads to the well-known functional used for rate-distortion optimized motion evaluation, as used for example in the JSVM encoder software. The Lagrangian multipliers λ_i are derived as in [20].

We additionally take into account the cost of the *enhancement* layer by also minimizing the enhancement layer distortion $D_1(\boldsymbol{p_1}|\boldsymbol{p_0})$ given the total bit rate $R_0(\boldsymbol{p_0})+R_1(\boldsymbol{p_1}|\boldsymbol{p_0})$ [21]. Weighting factor w is used to determine the trade-off between base layer and enhancement layer coding efficiency, leading to the cost functional

$$\begin{aligned}\min_{\boldsymbol{p_0},\boldsymbol{p_1}} \ & (1-w)\cdot(D_0(\boldsymbol{p_0}) + \lambda_0 R_0(\boldsymbol{p_0})) \\ & \qquad + w\cdot(D_1(\boldsymbol{p_1}|\boldsymbol{p_0}) + \lambda_1(R_0(\boldsymbol{p_0}) + R_1(\boldsymbol{p_1}|\boldsymbol{p_0}))).\end{aligned}$$

We examine the case where the motion information becomes identical to the information from the incoming bitstream when all layers are present in the SVC stream, i.e., no quality loss occurs after transcoding when no layers are dropped from the bitstream. As mentioned, this corresponds to a *data partitioning* scenario.

In this way, the distortion for the enhancement layer is eliminated, i.e., $D_1(\boldsymbol{p_1}|\boldsymbol{p_0}) = 0$, and the minimization problem becomes:

$$\min_{\boldsymbol{p_0},\boldsymbol{p_1}} \ (1-w)\cdot(D_0(\boldsymbol{p_0}) + \lambda_0 R_0(\boldsymbol{p_0})) + w\cdot\lambda_1(R_0(\boldsymbol{p_0}) + R_1(\boldsymbol{p_1}|\boldsymbol{p_0})).$$

For $w = 0$, the functional reduces to the case where no joint optimization is performed, i.e.,

$$\min_{\boldsymbol{p_0}} \ D_0(\boldsymbol{p_0}) + \lambda_0 R_0(\boldsymbol{p_0})$$

and only the base layer cost is minimized. In this case, base layer motion refinement will occur more frequently, since the cost of refinement bits is not taken into account. For $w = 1$, the expression

$$\min_{\boldsymbol{p_0}, \boldsymbol{p_1}} R_0(\boldsymbol{p_0}) + R_1(\boldsymbol{p_1}|\boldsymbol{p_0})$$

remains, under the side condition that reconstruction is identical when both layers are present in the bitstream. Typically, in this case, the optimum is achieved when all motion data is concentrated in the base layer, de facto corresponding to single-layer coding. Exceptions occur, however, when inter-layer motion prediction outperforms intra-layer median motion vector prediction. In these cases, identical motion data is obtained in the top layer at a reduced total bit rate, i.e., where the scalable stream (locally) outperforms the single-layer bitstream.

6. Results and discussion

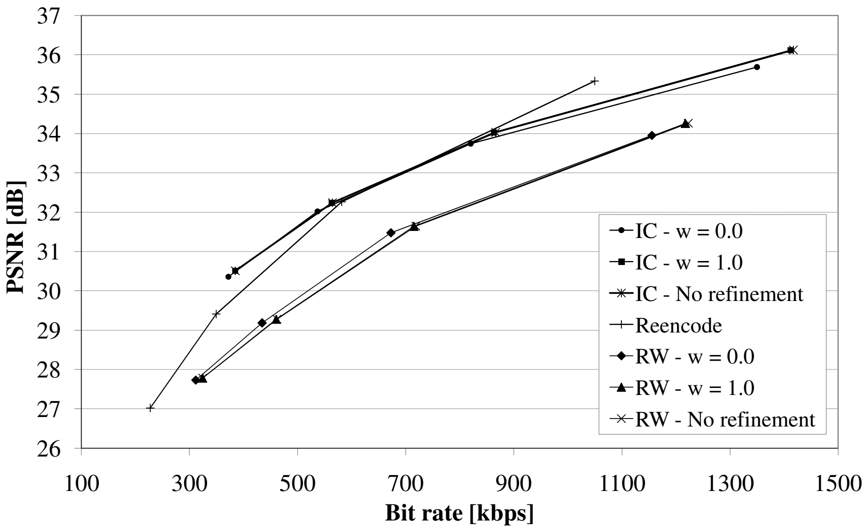
6.1. Implementation and setup

Several sequences (CIF and 4CIF resolution) were encoded using the Joint Model (single-layer) reference software, namely Foreman, Stefan, Mobile & Calender, Paris (CIF), and Soccer (4CIF). Hierarchical coding was used for the tests. A GOP length of 8 pictures was used and an intra period of 16. These single-layer bitstreams were transcoded using our transcoder software with and without motion refinement. We performed tests for two layers, i.e., one base layer and one enhancement layer. We used starting quantization parameters (QP_I) of 22, 27, 32, and 37. Note that, in order to avoid loss in residual data, we set these values equal to the quantization parameter QP_E of the output SVC enhancement layer, i.e., $QP_E = QP_I$. For the output base layer, a higher quantization parameter QP_B is set, i.e., $QP_B = QP_I + \Delta QP = QP_E + \Delta QP$. In order to cover typical use cases of SVC streams, we used ΔQP values of 6 and 12. In the tests, we focus on reduction of motion data in P and B pictures. Nonetheless, we give rate-distortion results for the overall streams, i.e., including intra-coded pictures without refinement.

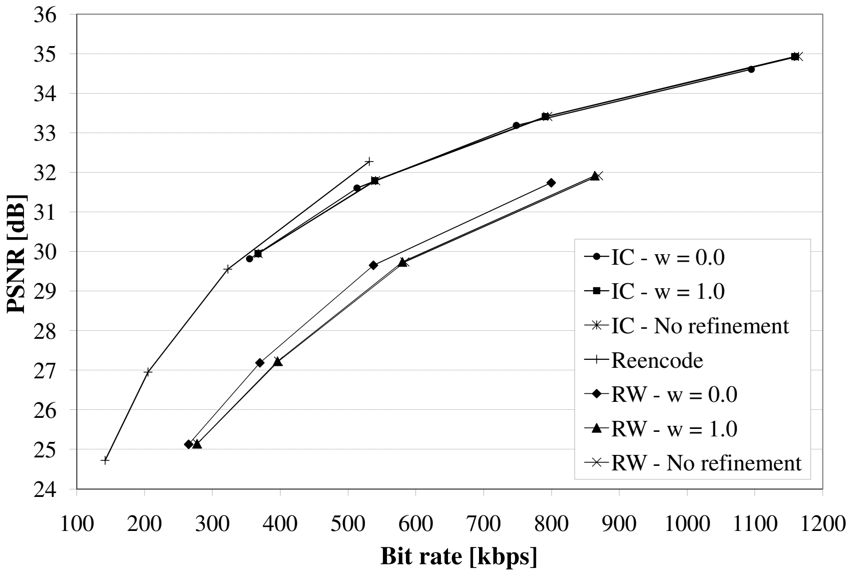
In the graphs, we make a distinction between the two architectures discussed in Sect. 2.5: both the intra copy (IC) and rewriting with spatial compensation (RW) architectures have been tested. For intra-coded pictures, the intra data is coded in the SVC base layer for the IC architecture. For the RW architecture, a decoding and re-encoding step is performed (with prediction mode reuse for the base layer), and inter-layer intra prediction is used for the enhancement layer. Given the low complexity of the intra prediction process, this re-encoding step can be performed with minimal impact on computational complexity. For intra-coded macroblocks in P and B pictures, the intra data is either copied to the base layer (IC architecture) or requantized with spatial compensation (RW architecture). Low-complexity open-loop rewriting is used in both cases for the motion-compensated macroblocks in P and B pictures. As a reference, we included the curves for reencoding in all graphs. To obtain these curves, the original streams were decoded and reencoded using the JSVM encoder.

6.2. Rate-distortion results

In Fig. 7(a), the rate-distortion results are shown for the base layer of the Stefan sequence, for $\Delta QP = 6$. For the RW case, the rate-distortion curve obtained by setting the enhancement layer weight to one (i.e., $w = 1.0$) practically coincides with the curve without motion refinement. By setting the enhancement layer weight to zero ($w = 0.0$), rate-distortion performance is improved by approximately 5%, in particular in the lower bit rate range. For the highest rate point, a reduction of the bit rate is found (by 5.5%, from 1223 kbps to 1155 kbps) at a marginal gain in rate-distortion performance (the curve is located marginally higher for the higher bit rate range). These results correspond with the theoretical model and illustrate that although distortion increases somewhat by merging partitions, the motion refinement model only allows a merge if the rate reduction is large enough to improve overall rate-distortion efficiency. The IC case results in rate-distortion curves with gains of more than 1 dB over the RW case. By concentrating the intra data in the base layer, a higher-quality base layer is obtained and drift is restricted, at the cost of an increased base layer bit rate. The motion refinement technique shows to be able to reduce the bit rate approximately along the RD curve, with a slight loss at the higher bit rate range (for $QP = 22$). It is clear that (in particular for the IC case) the model becomes less accurate when more residual data is present, and it becomes harder to estimate the distortion D. When compared to the reencoding curve, the IC curves obtain improved rate-distortion results in the lower bit rate range, but inferior results in the higher rate range. Lower bit rates are achieved using the RW rewriting case, at the cost of more than 1 dB PSNR loss. Although motion refinement offers a technique to reduce the bit rate of the streams after transcoding, and the IC rewriting technique is competitive with reencoding in the overlapping bit rate range, the full flexibility of reencoding will result in a base layer with a (significantly) lower bit rate.



(a) Results for Stefan sequence ($\Delta QP = 6$)).



(b) Results for Stefan sequence ($\Delta QP = 12$)).

Figure 7: Base layer R-D results for Stefan sequence ($\Delta QP = 6$ and $\Delta QP = 12$).

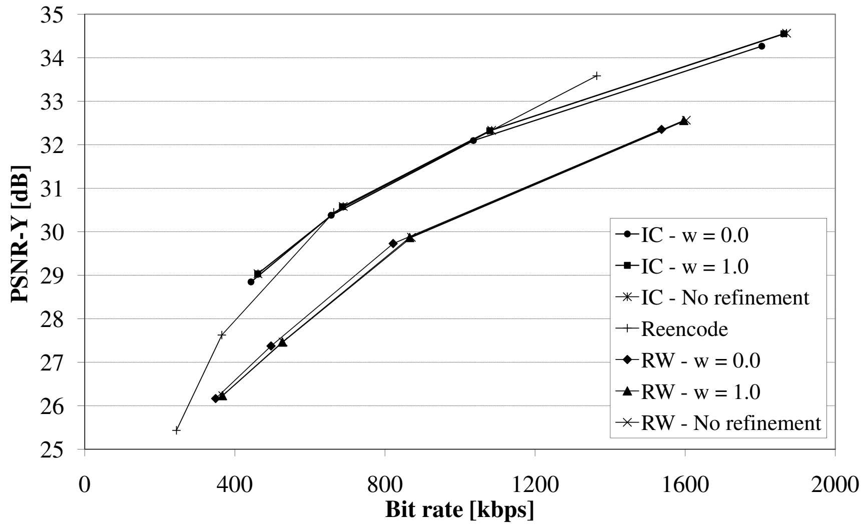
In Fig. 7(b), the results are shown for the same sequence, but with a

$\Delta QP = 12$ between the base and enhancement layer. As could be expected, a larger gap in quantization parameters (resulting in lower base layer bit rates) will lead to a higher degree of refined macroblocks in the stream. This leads to more potential for our motion-refined rewriting architecture, and gains of up to 0.5 dB for the RW curves. The base layer bit rates are reduced by 5% for the lower bit rate range to 8% for higher bit rates in this case. The same trends can be observed as in Fig. 7(a), albeit more pronounced. The IC curves obtain rate-distortion results competitive to the reencoding curve, but at significantly higher bit rates. Motion refinement is able to somewhat shift the curve towards the lower rate range. For RW rewriting, PSNR values are obtained comparable to those for reencoding, but at much higher bit rates.

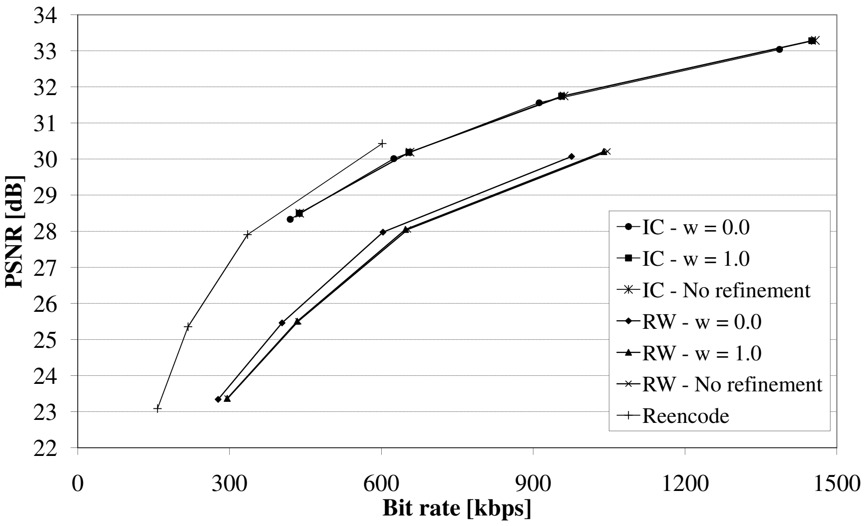
Similar results were obtained for the other sequences. The rate-distortion curves for the Mobile & Calender sequence are given in Fig. 8. The results were found to be comparable, with bit rate reductions of 6 to 8% for $\Delta QP = 12$ for the RW architecture. The performance of the IC architecture lies close to that of reencoding, but at higher bit rates.

Results for the Soccer sequence (4CIF resolution) are shown in Fig. 8. Rate-distortion gains are achieved for the entire bit rate range, with gains of approximately 5% for the higher bit rate range for the RW case. For this sequence, IC rewriting is able to clearly outperform reencoding, at the cost of higher base layer bit rates.

Results for the top layer are shown in Fig. 10. These curves illustrate another benefit of rewriting: both the IC and RW architecture are clearly able to outperform reencoding (by more than 1 dB). Also, the overhead of motion refinement becomes clear. Note that, since reconstruction is perfect in all cases (when compared to the original single-layer stream), identical PSNR values are obtained for all RD points at a given QP. Hence, only the corresponding rate values are of interest in these charts. For $w = 1.0$, no overhead is incurred when compared to the case where no refinement is used and both curves practically coincide. On the contrary, the total bit rate is even somewhat reduced (but for all sequences <1%). This is caused by cases where inter-layer motion vector prediction is more efficient than regular H.264/AVC inter-layer motion vector prediction. When the weight of the enhancement layer diminishes, the total bit rate will slowly increase, leading to the curves of $w = 0.5$ and $w = 0.0$. This increase in bit rate corresponds with the rate-distortion model, which states that for low values of w, the base layer rate-distortion performance behavior is optimized without taking into account the overall bit rate. The more merging operations are



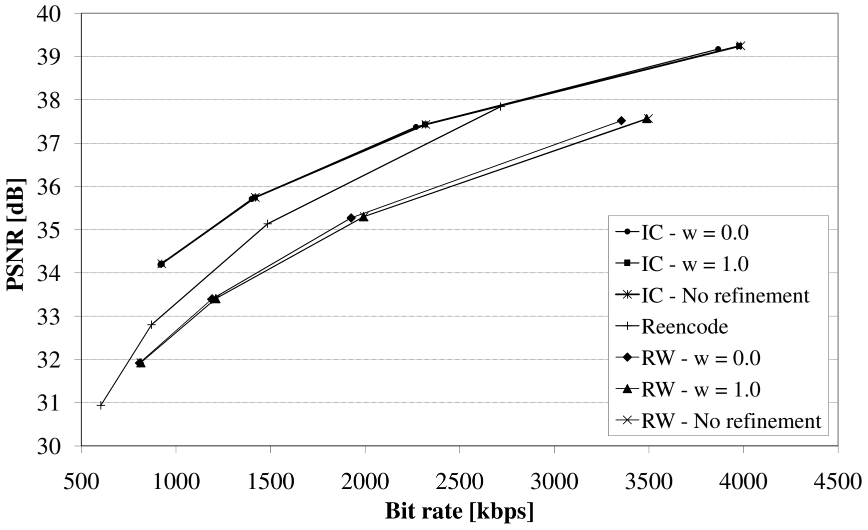
(a) Results for Mobile & Calender sequence ($\Delta QP = 6$)).



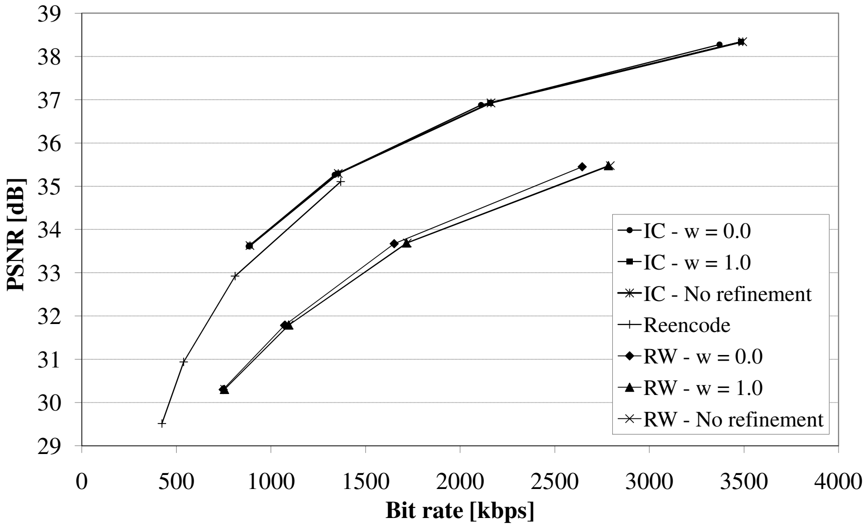
(b) Results for Mobile & Calender sequence ($\Delta QP = 12$)).

Figure 8: Base layer R-D results for Mobile & Calender sequence ($\Delta QP = 6$ and $\Delta QP = 12$).

performed in the base layer, the more information needs to be injected into the enhancement layer to reconstruct the original motion information. Since this introduces some redundancy in the bitstream (e.g., a macroblock type



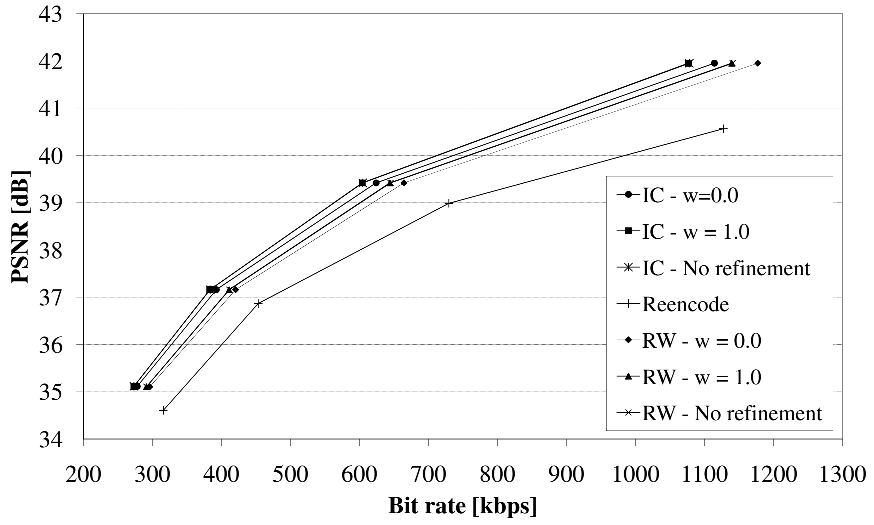
(a) Results for Soccer sequence ($\Delta QP = 6$)).



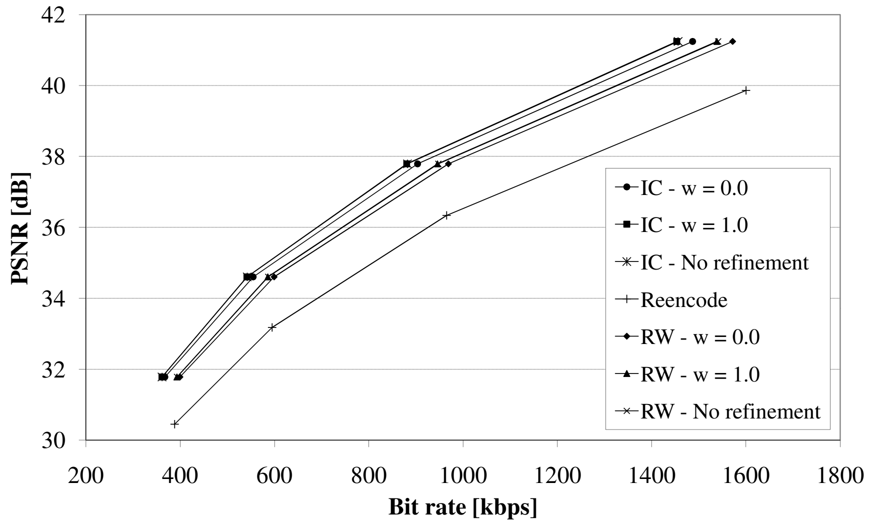
(b) Results for Soccer sequence ($\Delta QP = 12$)).

Figure 9: Base layer R-D results for Soccer sequence ($\Delta QP = 6$ and $\Delta QP = 12$).

syntax element needs to be sent in both layers in case of refinement), the overall bit rate will start to increase. The overhead of motion refinement in the overall bit rate, compared to the bit rate without motion refinement, is shown in Table 1 for the Foreman sequence and the RW architecture, for both $\Delta QP = 6$ and $\Delta QP = 12$. The small negative overhead for $w = 1.0$ can be noted in the bottom row of the table. For the IC architecture, higher bit rates are obtained for the overall bit rate, but the percentual overhead is very similar.



(a) Results for Foreman sequence ($\Delta QP = 6$)).



(b) Results for Paris sequence ($\Delta QP = 12$)).

Figure 10: Top-layer R-D results for Foreman and Paris sequences.

Table 1: Overhead of motion refinement in total bit rate (Foreman sequence, RW architecture, [%]).

	$\Delta QP = 6$				$\Delta QP = 12$			
	QP_I				QP_I			
w	22	27	32	37	22	27	32	37
0.0	3.19	3.01	2.17	1.64	3.41	3.16	2.21	1.65
0.2	3.06	2.95	2.15	1.64	3.39	3.16	2.21	1.65
0.4	2.86	2.56	1.71	1.33	3.39	3.16	2.21	1.65
0.6	2.07	1.88	1.19	0.78	3.33	3.11	2.19	1.64
0.8	0.24	0.25	0.18	0.14	3.07	2.59	1.60	1.03
1.0	-0.11	-0.13	-0.11	0.00	-0.12	-0.13	-0.12	0.01

6.3. Computational complexity results

In Table 2, average processing speed results are shown for rewriting with and without motion refinement. An Intel Xeon processor running at 2.66 GHz was used to generate the results. For the CIF resolution, our (non-optimized) transcoder implementation achieves results close to real time. It can be seen that the processing speed is reduced by 20 to 25% by adding motion refinement. The rewriting speed increases somewhat for higher QP values, given the reduced amount of residual data that needs to be processed. It is clear from the fifth row in the table that the proposed rewriting schemes significantly outperform reencoding, even when motion refinement is applied.

Table 2: Rewriting speed without and with motion refinement [fps]).

	CIF resolution				4CIF resolution			
	QP_I				QP_I			
	22	27	32	37	22	27	32	37
RW - without refinement	27.8	29.4	30.3	30.3	6.4	7.0	7.4	7.6
RW - with refinement	22.2	23.3	24.4	25.0	5.1	5.6	5.9	6.0
IC - without refinement	30.3	31.3	32.3	33.3	6.85	7.58	7.94	8.13
IC - with refinement	23.3	25.0	25.6	26.3	5.41	5.95	6.25	6.41
Reencoding	0.39	0.42	0.44	0.47	0.12	0.13	0.14	0.15

7. Conclusions

In this paper, we presented techniques for motion-refined rewriting of H.264/AVC streams to SVC. We introduced a multi-layer transcoder control algorithm that provides a trade-off in rate and distortion between the considered layers. By setting the weight factors appropriately, the model allows rate-distortion performance to be improved for the desired layer(s). Even though operations are performed entirely in the transform domain, we have shown that distortion caused by motion refinement is accurately taken into account in the model. Although additional distortion is introduced due to changes in the motion data, our approach intelligently decides whether or not refinement in the motion data should occur, leading to reduced base layer bit rates and an improvement in rate-distortion performance. In particular for the RW architecture, gains of up to 0.5 dB were obtained for the base layer. For the IC architecture, rate-distortion results are obtained that are competitive with reencoding, and motion refinement is able to reduce the base layer bit rate. The full flexibility of reencoding, however, will be able to produce a base layer with a lower bit rate, in particular for larger QP differences between the SVC layers. For the top layer, clear gains over reencoding are obtained for both presented architectures. Computational complexity results have shown that both rewriting architectures significantly outperform reencoding (by a factor of forty or more), both without and with motion refinement.

Acknowledgment

The research activities that have been described in this paper were funded by Ghent University, the Interdisciplinary Institute for Broadband Technology (IBBT), the Institute for the Promotion of Innovation by Science and Technology in Flanders (IWT-Flanders), the Fund for Scientific Research-Flanders (FWO-Flanders), and the European Union.

References

[1] ITU-T Rec. H.264 and ISO/IEC 14496-10 (MPEG-4 AVC), ITU-T and ISO/IEC JTC 1, Advanced Video Coding for Generic Audiovisual Services (Version 1: May 2003, Version 2: May 2004, Version 3: Mar. 2005, Version 4: Sept. 2005, Version 5 and Version 6: June 2006, Version 7: Apr. 2007, Version 8 (including SVC extension): November 2007).

[2] A. Vetro, C. Christopoulos, H. Sun, Video transcoding architectures and techniques: an overview, IEEE Signal Process. Mag. (2003) 18–29.

[3] J. Xin, C.-W. Lin, M.-T. Sun, Digital video transcoding, Proc. IEEE 93 (1) (2005) 84–97.

[4] I. Ahmad, X. Wei, Y. Sun, Y.-Q. Zhang, Video transcoding: an overview of various techniques and research issues, IEEE Trans. Multimedia 7 (5) (2005) 793–804.

[5] A. Eleftheriadis, P. Batra, Optimal data partitioning of MPEG-2 coded video, IEEE Trans. Circuits Syst. Video Technol. 14 (10) (2004) 1195–1209.

[6] A. Eleftheriadis, D. Anastassiou, Optimal data partitioning of MPEG-2 coded video, in: Proc. IEEE Int. Conf. Image Process. (ICIP), 1994.

[7] E. Barrau, MPEG video transcoding to a fine-granular scalable format, in: Proc. IEEE Int. Conf. Image Process. (ICIP), 2002.

[8] H. Shen, X. Sun, F. Wu, H. Li, S. Li, Transcoding to FGS streams from H.264/AVC hierarchical B-pictures, in: Proc. IEEE Int. Conf. Image Process. (ICIP), 2006.

[9] H. Schwarz, D. Marpe, T. Wiegand, Analysis of hierarchical B pictures and MCTF, in: Proc. IEEE Int. Conf. Multimedia Expo (ICME), 2006.

[10] J. De Cock, S. Notebaert, R. Van de Walle, A Novel Hybrid Requantization Transcoding Scheme for H.264/AVC, in: Proc. International Symposium on Signal Processing and its Applications (ISSPA 2007), 2007.

[11] J. De Cock, S. Notebaert, R. Van de Walle, Transcoding from H.264/AVC to SVC with CGS layers, in: Proc. IEEE Int. Conf. Image Process. (ICIP), 2007.

[12] A. Segall, SVC-to-AVC bit-stream rewriting for Coarse Grain Scalability, Joint Video Team, Doc. JVT-T061, Klagenfurt, Austria (July 2006).

[13] J. De Cock, S. Notebaert, P. Lambert, R. Van de Walle, Advanced bitstream rewriting from H.264/AVC to SVC, in: Proc. IEEE Int. Conf. Image Process. (ICIP), 2008.

[14] H. Schwarz, T. Wiegand, R-D optimized multi-layer encoder control for SVC, in: Proc. IEEE Int. Conf. Image Process. (ICIP), 2007.

[15] H. Schwarz, D. Marpe, T. Wiegand, Overview of the scalable video coding extension of the H.264/AVC standard, IEEE Trans. Circuits Syst. Video Technol. 17 (9) (2007) 1103–1120.

[16] H. Malvar, A. Hallapuro, M. Karczewicz, L. Kerofsky, Low-complexity transform and quantization in H.264/AVC, IEEE Trans. Circuits Syst. Video Technol. 13 (7) (2003) 598–603.

[17] S. Notebaert, J. De Cock, K. De Wolf, R. Van de Walle, Requantization transcoding of H.264/AVC bitstreams for Intra 4x4 prediction modes, in: Proc. Pacific-rim Conference on Multimedia (PCM), 2006.

[18] A. Secker, D. Taubman, Highly scalable video compression with scalable motion coding, IEEE Trans. Image Process. 13 (8) (2004) 1029–1041.

[19] H. Shen, X. Sun, F. Wu, Fast H.264/MPEG-4 AVC transcoding using power-spectrum-based rate-distortion optimization, IEEE Trans. Circuits Syst. Video Technol. 18 (6) (2008) 746–755.

[20] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, G. J. Sullivan, Rate-constrained coder control and comparison of video coding standards, IEEE Trans. Circuits Syst. Video Technol. 13 (7) (2003) 688–703.

[21] K. Ramchandran, A. Ortega, M. Vetterli, Bit allocation for dependent quantization with applications to multiresolution and MPEG video coders, IEEE Trans. Image Process. 3 (5) (1994) 533–545.