Proceedings of the 2007 IEEE Symposium on Computational
Intelligence in Image and Signal Processing (CIISP 2007)

Improved H.264/AVC Requantization Transcoding using Low-Complexity Interpolation Filters for 1/4-Pixel Motion Compensation

Stijn Notebaert, Jan De Cock, and Rik Van de Walle
Ghent University – IBBT
Department of Electronics and Information Systems – Multimedia Lab
Gaston Crommenlaan 8 bus 201, B-9050 Ledeberg-Ghent, Belgium
Email: {stijn.notebaert, jan.decock, rik.vandewalle}@ugent.be

Abstract— **Transcoding is a fast technique for video adaptation, which offers great flexibility for changing the characteristics of already coded video sequences in order to match the properties of the terminal display devices and the bandwidth availability of the transmission networks. For reduction of the bitrate, requantization transcoding is often used. In our previous work, we introduced a hybrid architecture for requantization transcoding, which is able to reduce the bitrate for H.264/AVC bitstreams with quality results approaching the rate-distortion optimal decoder-encoder cascade. In this paper, we focus on the filters used for interpolation of the accumulated requantization errors, which are used for compensation of the current (sub-)macroblock partition. We show that by using reduced complexity filters, computational complexity can be significantly reduced without quality loss.**

I. Introduction

The concept of Universal Multimedia Access establishes the possibility to access multimedia content by any terminal through any network. In order to meet the constraints imposed by the plethora of terminals and networks, adaptation of the video content is essential. One possible adaptation consists in the reduction of the bitrate by decreasing the amount of residual data present in the video bitstream. The objective of bitrate reduction is to reduce the bitrate while maintaining low complexity and achieving the highest quality possible [1], [2]. Ideally, the reduced bitstream should have the quality of the bitstream directly generated at the reduced bitrate.

Transcoding is a technique which offers elegant and efficient adaptation of already coded video bitstreams. Different types of transcoding algorithms exist which allow rate reduction of the bitstreams. One important technique for bitrate reduction consists in requantization of the residual data with a coarser quantization parameter (QP). In the past, efficient requantization techniques have been succesfully applied on MPEG-2 bitstreams [3], [4], [5]. In [3], the authors presented various transcoding techniques for bitrate reduction of MPEG-2 bitstreams, with different implementation complexity and associated trade-offs in the resulting video quality. The authors of [4] proposed a drift-free MPEG-2 video transcoder, working entirely in the frequency domain. A theoretical analysis of the transcoding problem of MPEG-2 intra predicted frames completed with the derivation of new quantization methods for efficient transcoding was given by Werner in [5].

The improved compression efficiency of H.264/AVC over MPEG-2 results in an increased number of dependencies in the coded bitstream [6]. This has a serious impact on the requantization algorithms in order to guarantee transcoded bitstreams with acceptable quality. An architecture for requantization transcoding of H.264/AVC bitstreams was proposed in [7], using techniques based on MPEG-2 requantization transcoding. The authors assessed the requantization errors of H.264/AVC bitstreams, caused by requantized values which are used for intra prediction and motion compensation. For motion-compensated macroblocks in P and B slices, a compensation was used based on the accumulated requantization errors from previous reference frames. The authors did not, however, provide a solution to the problem of transcoding of intra-predicted macroblocks in P and B slices. This resulted in an architecture with varying quality results depending on the characteristics of the incoming bitstream, notably by the amount of intra-predicted macroblocks present.

In [8], [9], we proposed new algorithms which compensate for the errors introduced by the requantization process in intra-predicted macroblocks. This results in significantly improved visual quality and drift reduction. In [10], we combined our techniques for intra-predicted macroblocks with algorithms for inter-predicted pictures, and obtained a hybrid requantization transcoding architecture with quality results that are independent of the characteristics of the video sequence.

Because the H.264/AVC specification uses motion compensation with fractional-pixel displacement (half-pixel and quarter-pixel accuracy), it is necessary to interpolate the accumulated requantization errors in order to obtain a meaningful compensation for the current (sub-)macroblock partition. In our previous work we used the standard H.264/AVC 6-tap interpolation filter. The question remains if it possible to further reduce the computational complexity of the transcoder architecture by using low-complexity interpolation filters, without harming the visual quality of the output bitstreams.

The remainder of the paper is as follows: in Sect. II, we give a brief overview of the hybrid architecture of the H.264/AVC requantization transcoder. The different interpolation filters

1-4244-0707-9/07/$25.00 ©2007 IEEE

are presented and discussed in Sect III. Section IV gives experimental results of the requantization transcoder using the presented interpolation filters. Finally, the paper is concluded in Sect. V.

II. Requantization architecture

In [10], we proposed a hybrid architecture for requantization transcoding that offers fast computational performance and good objective and subjective quality results. Depending on the slice and the macroblock type being considered, a different transcoding technique is used for requantization. The different techniques which are exploited for intra-predicted and motion-compensated pictures are discussed in the remainder of this section. An overview of the requantization techniques for different slice and macroblock types is given in Table I.

TABLE I
REQUANTIZATION TECHNIQUES FOR DIFFERENT SLICE AND MACROBLOCK TYPES IN THE HYBRID ARCHITECTURE.

Slice type	MB type	Requantization technique
I	I	mode reuse
P	I	transform-domain compensation (S-TDC)
P	P	transform-domain compensation (T-TDC)
B	I	transform-domain compensation (S-TDC)
B	P, B	open-loop

A. Requantization of intra-predicted pictures

Since intra-predicted pictures are used as a further reference for motion-compensated pictures, the quality of the intra-predicted pictures has to be high to limit further degradation of the video. For optimum quality we use a transcoder with mode reuse for the intra-predicted pictures. In [8], [9], we showed that this architecture gives quality results close the optimal decoder-encoder cascade. When compared to transform-domain architectures, drift is reduced that may arise due to non-linearities such as the downscaling in the inverse H.264/-AVC integer transform and the absence of clipping operations in transform-domain intra prediction. As a downside, two coding loops and two frame buffers are required, one in the decoder loop, and another in the encoder loop. The transcoder with mode reuse results in reference frames with high quality. This type of transcoder is shown in Fig. 1, where the symbol $\Im_p$ indicates the intra prediction.

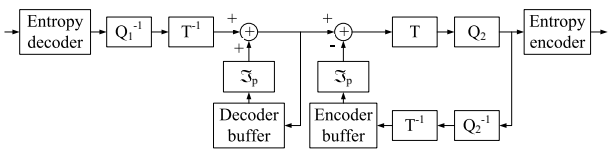

Fig. 1. Mode reuse transcoder for intra-predicted pictures.

B. Requantization of motion-compensated pictures

For motion-compensated pictures, we use a hybrid architecture, which combines transcoding techniques for intra-predicted and motion-compensated macroblocks. This architecture is depicted in Fig. 2.

For P type macroblocks in P slices, transform-domain compensation is applied, using temporally accumulated requantization error values (T-TDC). The compensation matrix Φ is obtained by motion compensation with the interpolated accumulated requantization errors from previous reference frames. [1]

For I type macroblocks in P and B slices, we use the architecture with transform-domain compensation, here using spatially accumulated requantization error values (S-TDC). As described in our previous work [8], [9], the intra prediction formulas and the forward integer transform can be combined to obtain sparse compensation matrices Φ, resulting in a low-complexity architecture.

An essential requirement for the hybrid architecture is that two frame buffers are used. The first frame buffer contains the requantization errors from the previous reference frame, which will be used for motion compensation of the block being processed. After each block is processed, the results are stored in the second buffer. In this way, the requantization errors of the current frame can be used for intra compensation of surrounding 4×4 or 16×16 blocks. Hence, the pixel buffer used depends on the current macroblock type (motion compensated or intra predicted).

In order to obtain the correct values after requantization of the first (intra-predicted) picture in a GOP, the first buffer is filled with the differences of the decoder and encoder buffer of the mode reuse architecture. This results from the fact that in the mode reuse architecture, the pixel values are reconstructed, while in the hybrid architecture, only the requantization differences are stored.

C. Transcoder performance

In order to illustrate the transcoding performance of the hybrid architecture we briefly discuss the rate-distortion curves for transcoded H.264/AVC bitstreams. The H.264/AVC bitstreams (CIF resolution) were coded using an IBBP GOP-structure of 30 frames. The R-D performance of the Stefan sequence is shown in Fig. 3. As can be seen from this R-D curve, the PSNR gap between the rate-distortion optimal decoder-encoder cascade and our approach is limited for low bitrates, but for higher bitrates the PSNR gap increases. The R-D curve clearly branches away from the optimal curve for higher bitrates, due to non-linear operations in the interpolation process, on the one hand, and the open-loop requantization of motion-compensated macroblocks in B slices, on the other

[1] Note that there is a clear distinction between 'compensation', by which we mean an adjustment for the accumulated requantization errors in the transcoder, and the 'motion compensation' which is present in the encoder and decoder, as defined in the H.264/AVC specification. In the remainder of this paper, we always refer to the latter using the exact phrase 'motion compensation'.

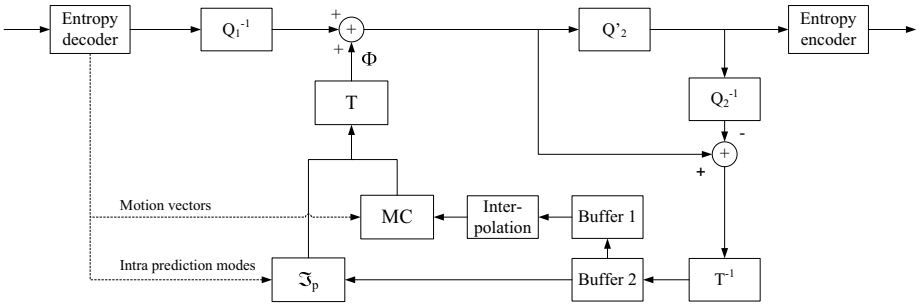

Fig. 2. Hybrid architecture for motion-compensated pictures.

hand. Since the B pictures in the IBBP GOP structure are not used as a reference for further prediction, however, the open-loop requantization does not introduce drift in the transcoded sequence.

In [7], a mixed requantization architecture (MRA) was proposed for requantization transcoding. As mentioned earlier, the authors did not provide a solution for intra-predicted macroblocks in P and B slices.

Compared to the MRA architecture, the introduction of compensation methods for intra-predicted macroblocks in our hybrid architecture results in noticeable quality gains over the complete range of bitrates, as can be seen in in Fig. 3.

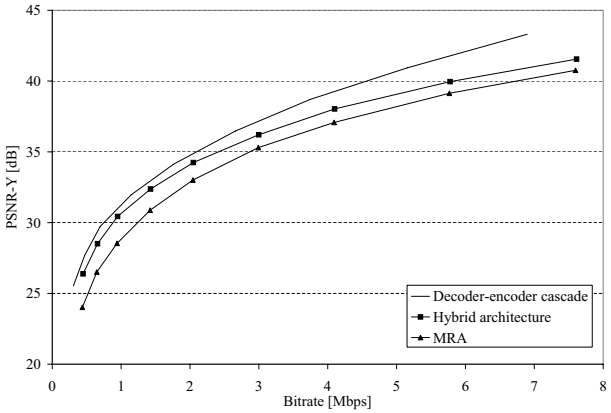

Fig. 3. R-D performance for quarter-pixel transcoding (Stefan).

Figure 4 illustrates that the PSNR loss is to a large extent caused by the fractional-pixel motion prediction, as the performance for transcoding of full-pixel motion-compensated sequences achieves the rate-distortion performance of the decoder-encoder cascade when no B slices are present.

In the architecture described above, we used the standard H.264/AVC interpolation filter. However, taking into account the high computational complexity of the H.264/AVC 6-tap filter, we could try to determine a reduced complexity interpolation filter for our transcoding architecture. In order to find the best matching filter for our transcoding problem, we compare different interpolation filters concerning computational complexity of the filter implementations and the visual quality of

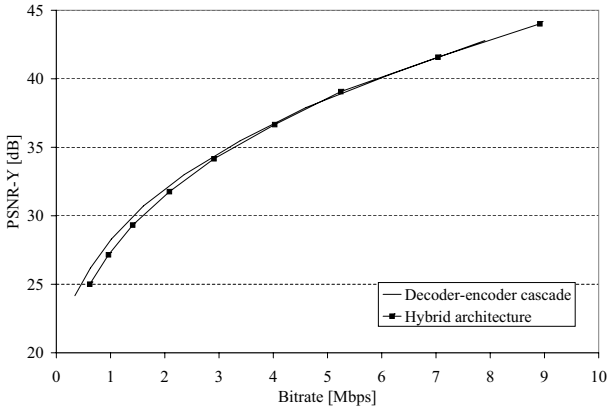

Fig. 4. R-D performance for full-pixel transcoding (Stefan).

the transcoded bitstreams.

III. Interpolation filters

Because motion vectors in H.264/AVC use quarter-pixel accuracy, an essential question is how to use the quantization error values in order to compensate for the pixels in the current picture. Since full-pixel reference values are interpolated at the encoder and decoder, the requantization error values also need to be interpolated to get a quarter-pixel interpolation value, according to the current motion vector. In [10], we used the H.264/AVC 6-tap interpolation filter as a standard way to interpolate the requantization error values. In the following sections, however, we show that a number of alternative filters can be used that provide a significant reduction in computational complexity at a minimal loss in visual quality.

A. H.264/AVC 6-tap filter

In H.264/AVC, motion compensation is performed with quarter-pixel accuracy [11]. In order to obtain the interpolated pixel values, a 6-tap filter is first used. The kernel $(1, -5, 20, 20, -5, 1)/32$ is applied in both horizontal and vertical direction, resulting in the half-pixel values. The quarter-pixel values are obtained by succeedingly averaging these values, using two surrounding full- or half-pixel values. These steps are illustrated in Fig. 5 and Fig. 6, respectively.

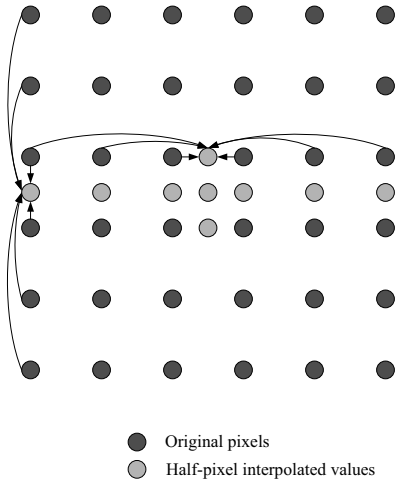

Fig. 5. First step of the H.264/AVC 6-tap interpolation process.

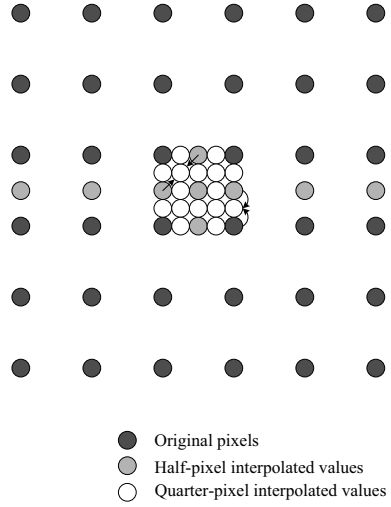

Fig. 6. Second step of the H.264/AVC 6-tap interpolation process.

Because the 6-tap filter has to be applied several times for every pixel (on average about 3 times), and (mainly) because of the multiplications present, it is clear that quarter-pixel interpolation is a computationally complex process. Reduction of computational complexity is a major driving force in the development of transcoding solutions. Hence, lower complexity solutions are needed that allow fast transcoding without compromising visual quality.

B. Nearest neighbor

Obviously, not using interpolation, and hence using the full-pixel requantization values, is the solution with minimal complexity. Since the incoming motion vectors have quarter-pixel accuracy, these have to be adapted to full-pixel accuracy. One possibility is to round the motion vectors in both directions, hence only keeping the integer part.

```
xInt=xA+(mvX+2)>>2
yInt=yA+(mvY+2)>>2,
```

where (xA,yA) denotes the position of the upper-left sample of the current (sub-)macroblock partition, (xInt,yInt) represents the position in full-sample units in the reference picture, and (mvX,mvY) is the motion vector of the current (sub-)macroblock partition.

This corresponds to a mapping of 16 possible motion vectors to one, as can be seen in Fig. 7. As can be seen, this corresponds to the concept of a nearest neighbor filter. Note that in the nearest neighbor case, the computational complexity only depends on the number of motion vector predicted (sub-)macroblock partitions in the currently processed picture.

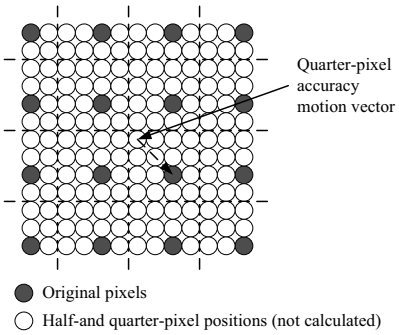

Fig. 7. Mapping of motion vectors (nearest neighbor).

C. Bilinear filter

As a second alternative, we propose to use a traditional bilinear filter $(1,1)/2$, applied to obtain both the half-pixel as well as the quarter-pixel interpolated values. The interpolation process is shown in Fig. 8. The second step of the interpolation process is identical to that of the H.264/AVC 6-tap filter. Apart from the reduced amount of memory operations, this approach requires only additions and shifts. Hence, using a bilinear filter implies a large computational gain when compared to the 6-tap filter.

D. Computational complexity

A summary of the computational complexity of the three methods is shown in Table II, along with the required number of executions per picture for a CIF sequence. These numbers are obtained by taking into account that for an $M \times N$ picture, $3MN - 2M - 2N + 1$ half-pixel values have to be calculated, along with $12MN - 10M - 10N + 8$ quarter-pixel values. As mentioned, the calculations for the nearest-neighbor technique do not depend on the size of the picture, but on the amount of motion vector predicted (sub-)macroblock partitions, and thus on the amount of motion in the video sequence.

The table indicates that by using a bilinear filter instead of the H.264/AVC 6-tap filter, more than $600,000$ multiplications can be saved per CIF picture. In the next section, we show the implications of using different filters on the visual quality of the sequence.

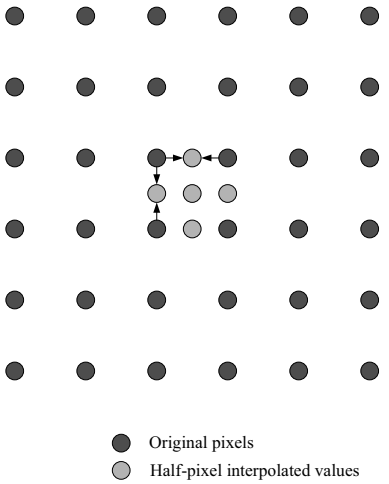

Fig. 8. First step of the Bilinear interpolation process.

TABLE II
COMPUTATIONAL COMPLEXITY OF THE FILTERS.

Filter	First step			Second step		
	+	*	>>	+	*	>>
H.264/AVC 6-tap	6	2	1	2	0	1
Bilinear	2	0	1	2	0	1
Number of executions	$302,849$			$1,210,120$		

IV. EXPERIMENTAL RESULTS

In this section, we show the results that were obtained by using the hybrid transcoding architecture, when applying different filters for interpolation of requantization error values. We show results for the standard H.264/AVC interpolation filter, and compare them to results obtained through the reduced complexity bilinear and nearest neighbor filters. For the tests, we encoded several test sequences, in CIF format, using the Joint Model reference software, version 11.0. The sequences were then transcoded using our transcoder implementation. In the following section, we show the results for sequences with varying motion characteristics, namely Stefan, Foreman, and Paris.

For Stefan, the results are shown in Fig. 9. From the rate-distortion curve, it is clear that there are only minor differences between the results for the 6-tap and bilinear filter. The motion vector rounding, or nearest neighbor filter performs slightly worse, with a PSNR gap of about 0.5 dB. The results indicate that a serious reduction in computational complexity can be achieved by replacing the H.264/AVC interpolation filter by a fast bilinear filter, without significantly compromising the PSNR results. More detailed results can be found in Table III for the Stefan sequence.

Similar results are obtained for the Foreman and Paris sequences, as can be seen in Fig. 10 and Fig. 11. It can be remarked that for very high bitrates, the bilinear filter performs better than the H.264/AVC 6-tap filter. This can be explained that at low quantization parameters, the quantization intervals and hence the quantization errors become increasingly small.

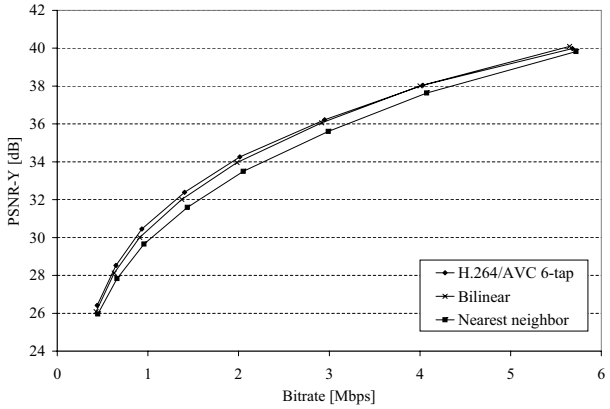

Fig. 9. R-D performance (Stefan sequence).

TABLE III
DETAILED PSNR VALUES FOR DIFFERENT FILTERS (STEFAN SEQUENCE).

H.264/AVC 6-tap		Nearest neighbor		Bilinear	
Bitrate [Mbps]	PSNR-Y [dB]	Bitrate [Mbps]	PSNR-Y [dB]	Bitrate [Mbps]	PSNR-Y [dB]
5.68	39.98	5.72	39.82	5.65	40.10
4.03	38.05	4.07	37.63	4.00	38.01
2.95	36.22	2.99	35.61	2.92	36.06
2.01	34.26	2.05	33.50	1.98	33.96
1.40	32.39	1.44	31.60	1.37	32.02
0.93	30.45	0.96	29.65	0.91	30.00
0.65	28.53	0.66	27.83	0.63	28.13
0.44	26.42	0.45	25.96	0.43	26.09

Hence, the interpolated values are to a large extent determined by noise resulting from the non-linear operations in the compensation loop. In this case, it is not beneficial to combine a large number of quantization errors, as is the case for the H.264/AVC 6-tap filter.

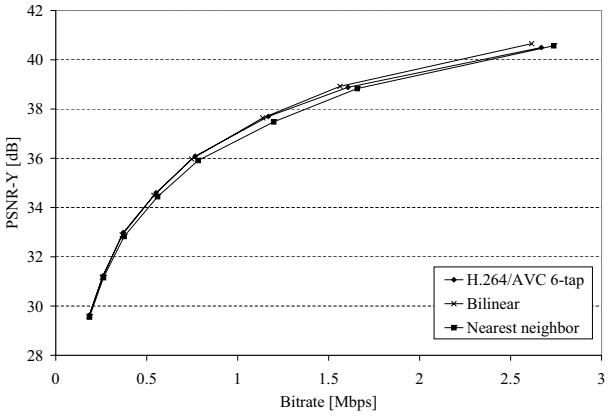

Fig. 10. R-D performance (Foreman sequence).

A frame-per-frame visualization of the PSNR values for the first 90 frames of the Stefan sequence is given in Fig. 12. Here we can see that the quality slowly degrades due to non-linearities in the requantization transcoding loop. We can clearly distinguish the B pictures, which have a slightly

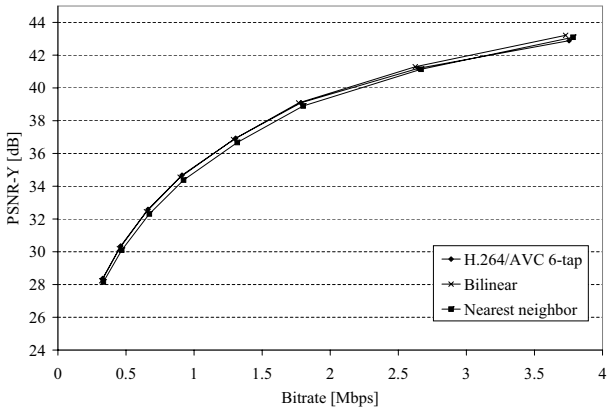

Fig. 11. R-D performance (Paris sequence).

lower PSNR due to the open-loop requantization transcoding in these pictures. This, however, does not affect the quality of the remaining pictures in the GOP, since they are not used as reference in the IBBP GOP structure. The quality of the H.264/AVC 6-tap transcoded pictures and the bilinear filter transcoded pictures is practically identical, while the results for the nearest neighbor filter slowly diverge, resulting in a PSNR gap of more than 1 dB at the end op the GOP.

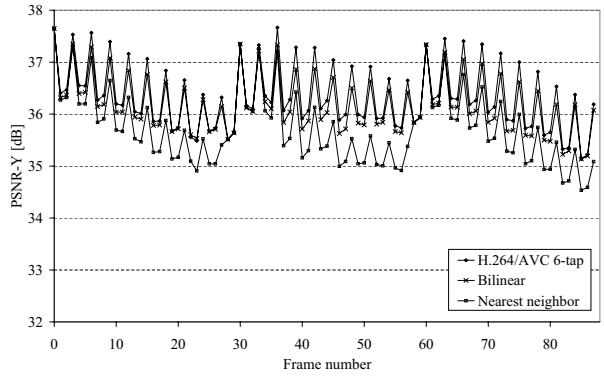

Fig. 12. R-D performance (Stefan sequence).

Table IV shows the percentage of the transcoding time of the bilinear and nearest neighbor implementations relative to the requantization transcoder with standard H.264/AVC interpolation. As can be seen, serious reductions in execution time can be obtained by using the reduced complexity interpolation filters. Gains up to 23% are obtained for the bilinear filter, and up to 38% for the nearest neighbor implementation.

TABLE IV
RELATIVE TRANSCODING TIME.

Sequence	Bilinear (%)	Nearest neighbor (%)
Stefan	81.5	68.5
Foreman	80.6	66.1
Paris	76.9	61.4

V. CONCLUSIONS

Requantization transcoding is a fast and elegant technique for bitrate reduction. In this paper, we refined our previously introduced hybrid architecture for requantization transcoding of H.264/AVC bitstreams. We examined the quarter-pixel interpolation process, which is required for interpolation of the accumulated requantization errors in order to obtain a meaningful compensation of the currently processed picture. We showed that a significant reduction in computational complexity can be obtained by using a bilinear filter instead of the standard H.264/AVC 6-tap filter, with no losses in PSNR. An even higher reduction in computational complexity can be obtained by using a nearest neighbor filter, which can be implemented by rounding off the motion vectors to full-pixel accuracy. This type of filter can be used if a minor loss in visual quality is tolerated.

VI. ACKNOWLEDGEMENTS

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), the Fund for Scientific Research-Flanders (FWO-Flanders), the Belgian Federal Science Policy Office (BFSPO), and the European Union.

REFERENCES

[1] Anthony Vetro, Charilaos Christopoulos, and Huifang Sun. Video transcoding architectures and techniques: an overview. *IEEE Signal Processing Magazine*, pages 18–29, March 2003.

[2] Ishfaq Ahmad, Xiaohui Wei, Yu Sun, and Ya-Qin Zhang. Video transcoding: an overview of various techniques and research issues. *IEEE Transactions on Multimedia*, 7(5):793–804, October 2005.

[3] Huifang Sun, Wilson Kwok, and Joel Zdepski. Architectures for MPEG compressed bitstream scaling. *IEEE Transactions on Circuits and Systems for Video Technology*, 6(2):191–199, April 1996.

[4] Pedro Assunção and Mohammed Ghanbari. A frequency-domain video transcoder for dynamic bit-rate reduction of MPEG-2 bit streams. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(8):953–967, December 1998.

[5] Oliver Werner. Requantization for transcoding of MPEG-2 intraframes. *IEEE Transactions on Image Processing*, 8(2):179–191, February 1999.

[6] Thomas Wiegand, Gary Sullivan, Gisle Bjøntegaard, and Ajay Luthra. Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):560–576, July 2003.

[7] Damien Lefol, Dave Bull, and Nishan Canagarajah. Performance evaluation of transcoding algorithms for H.264. *IEEE Transactions on Consumer Electronics*, 52(1):215–222, February 2006.

[8] Jan De Cock, Stijn Notebaert, Peter Lambert, Davy De Schrijver, and Rik Van de Walle. Requantization transcoding in pixel and frequency domain for intra 16x16 in H.264/AVC. In *Proceedings of ACIVS 2006 (Advanced Concepts for Intelligent Vision Systems)*, 2006.

[9] Stijn Notebaert, Jan De Cock, Koen De Wolf, and Rik Van de Walle. Requantization transcoding of H.264/AVC bitstreams for intra 4x4 prediction modes. In *Proceedings of PCM 2006 (Pacific-rim Conference on Multimedia)*, 2006.

[10] Jan De Cock, Stijn Notebaert, and Rik Van de Walle. A novel hybrid requantization transcoding scheme for H.264/AVC. In *Proceedings of ISSPA 2007 (International Symposium on Signal Processing and its Applications)*, 2007. Accepted for publication.

[11] Gisle Bjøntegaard. Motion compensation with 1/4 pixel accuracy. *ITU-T SG16 Q.15*, February 2000.