Requantization Transcoding of H.264/AVC Bitstreams for Intra 4×4 Prediction Modes

Stijn Notebaert, Jan De Cock, Koen De Wolf, 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
{stijn.notebaert, jan.decock, koen.dewolf, rik.vandewalle}@ugent.be
http://multimedialab.elis.ugent.be

Abstract. Efficient bitrate reduction of video content is necessary in order to satisfy the different constraints imposed by decoding devices and transmission networks. Requantization is a fast technique for bitrate reduction, and has been successfully applied for MPEG-2 bitstreams. Because of the newly introduced intra prediction in H.264/AVC, the existing techniques are rendered useless. In this paper we examine requantization transcoding of H.264/AVC bitstreams, focusing on the intra 4×4 prediction modes. Two architectures are proposed, one in the pixel domain and the other in the frequency domain, that compensate the drift introduced by the requantization of intra 4×4 predicted blocks. Experimental results show that these architectures perform approximately equally well as the full decode and recode architecture for low to medium bitrates. Because of the reduced computational complexity of these architectures, in particular the frequency-domain compensation architecture, they are highly suitable for real-time adaptation of video content.

Keywords: bitrate reduction, H.264/AVC, requantization, transcoding.

1 Introduction

More and more video content is coded using the state-of-the-art H.264/AVC video coding standard. This content has to be made available to a large number of devices with varying network characteristics and network connectivity. In order to meet the different constraints, elegant solutions are needed for fast adaptation of the video content to a lower bitrate. Architectures for bitrate adaptation have been discussed in the past, more specifically for MPEG-2 coded video content [1,2]. Requantization transcoding is a fast technique for bitrate reduction, and is able to approximate the quality of a full decoder-recoder with large reduction in computational complexity [3]. The problem of requantization of intra-coded frames in MPEG-2 has been examined in [4].

Due to the increased complexity of the encoding process and the newly introduced intra prediction in H.264/AVC, these results no longer apply as such. The augmented number of dependencies in the coded pictures no longer allow straightforward open-loop requantization.

Since intra prediction in H.264/AVC is performed on 4×4 and 16×16 , a clear distinction has to be made between both. We have described requantization techniques for intra 16×16 prediction in [5], hence this discussion is omitted here. In this paper, we tackle the problem of requantization for the intra 4×4 prediction modes of H.264/AVC, and propose two architectures that compensate the errors due to drift propagation, resulting in highly improved visual quality of the transcoded stream.

The remainder of this paper is organized as follows. An overview of the relevant H.264/AVC coding tools is given in Sect. 2. In Sect. 3, we revisit the open-loop requantization architecture, applying it to H.264/AVC, and introduce two drift-compensating architectures. The results of the implementation of the transcoders are given in Sect. 4. Finally, conclusions are given in Sect. 5.

2 H.264/AVC Tools

2.1 Intra Prediction

Intra prediction is used to exploit the spatial redundancy between neighbouring pixels. A block is predicted using previously encoded and reconstructed pixels of surrounding blocks. In H.264/AVC, a macroblock can be predicted using a combination of nine 4×4 or one of four 16×16 intra prediction modes. The intra prediction, which was not present in, for example, MPEG-2 Video, results in an improved compression efficiency. However, it also introduces a number of dependencies. As we will see, this has an important impact on the perceptual quality of the transcoded video sequences. In this paper, we focus on the intra 4×4 prediction.

2.2 H.264/AVC Transform and Quantization

The integer transform in the H.264/AVC specification [6,7] is based on the Discrete Cosine Transform, and is applied on 4×4 blocks. The forward transform of a 4×4 block X is represented by

$$Y = (C_F X C_F^T) \otimes E_F \ ,$$

where C_F represents the kernel transformation matrix. E_F is the post-scaling matrix. For efficiency reasons, the post-scaling operation of the transformation is postponed and integrated in the quantization process. After the core transformation $W_{ij} = (C_F X C_F^T)_{ij}$, with $i, j = 0, \ldots, 3$, the coefficients W_{ij} are quantized. H.264/AVC provides 52 values for the Quantization Parameter (QP), which can vary on a macroblock basis. The values of QP were defined in such a way that, if QP is increased with a value of 6, the quantization step is doubled and the bitrate is approximately halved. This non-linear behaviour results in the possibility to target a broad range of bitrates. The forward quantization can be implemented as

$$|Z_{ij}| = (|W_{ij}| \cdot M_{ij} + f) \gg qbits$$

where $qbits = 15 + \lfloor QP/6 \rfloor$, and f represents the dead zone control parameter [6]. The multiplication factor M_{ij} is determined by QP mod 6 and the position in the 4×4 block. At the decoder side, the process is defined as follows. The inverse quantization process is defined as

$$W'_{ij} = Z_{ij} \cdot V_{ij} \cdot 2^{\lfloor QP/6 \rfloor} .$$

The values of M_{ij} and V_{ij} result in the coefficients W'_{ij} that exceed the pre-quantized values W_{ij} by a factor $64 \cdot E_{F_{ij}} \cdot E_{I_{ij}}$, hence including the post-scaling of the forward transform along with the pre-scaling of the inverse transform:

$$X' = C_I^T (Y \otimes E_I) C_I .$$

The factor 64 is introduced to avoid rounding errors in the inverse transformation that follows. We refer to [6,7] for more information about the intertwined transformation and quantization.

3 Requantization Transcoder Architectures

3.1 Open-Loop Transcoder

The most straightforward method for requantization is the open-loop requantization transcoder, as shown in Fig. 1. This type of transcoder consists of a dequantization step Q_1^{-1}, followed by a requantization step Q'_2 with a coarser quantization parameter (QP).

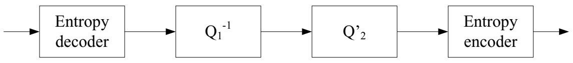

Fig. 1. Open-loop requantization transcoder

The implementation of this type of requantization transcoder was rather straightforward in MPEG-2 [4]. However, in H.264/AVC, special attention has to be paid to the requantization Q'_2. The multiplication factors have to be adapted in order to bring into account the scaling factors $E_{F_{ij}}$ and $E_{I_{ij}}$ of the H.264/AVC integer transform [5,8]. Since these scaling factors are already applied in the original quantization, they may not be repeated in the requantization. As a result, the multiplication factors for the integer transformation have to be downscaled by the factors 4, 2.56 and 3.2, depending on their position (i, j) in the 4×4 block of coefficients. The downscaling factors arise from:

$$(M_{ij} \cdot V_{ij}) \gg 15 = 64 \cdot E_{F_{ij}} \cdot E_{I_{ij}} = \begin{cases} 4, & r = 0 \\ 2.56, & r = 1 \\ 3.2, & r = 2 \end{cases}$$

where the factor 64 is introduced to avoid rounding errors during the inverse transform[1], and

$$r = \begin{cases} 0, & (i,j) \in \{(0,0),(0,2),(2,0),(2,2)\} \\ 1, & (i,j) \in \{(1,1),(1,3),(3,1),(3,3)\} \\ 2, & \text{otherwise} \end{cases}$$

The quality loss due to open-loop requantization in MPEG-2 intra frames remains limited, and is caused only by requantization noise [3]. In H.264/AVC, due to the introduction of intra prediction, drift arises and accumulates the requantization differences throughout the pictures. As will be seen in Sect. 4, this results in unacceptable quality of the transcoded video.

3.2 Requantization Transcoder with Pixel-Domain Compensation (PDC)

In order to avoid drift, a full decoder-recoder (FDR) might be used. The resulting sequence will have the highest achievable quality, but clearly, this concatenation will be too time-consuming to be used in real-time adaptation engines. An important bottleneck consists in searching the optimal prediction mode. This is especially true for rate-distortion optimized mode search, which includes entropy coding of every examined prediction mode. An alternative is to re-use the coding mode decisions from the incoming bitstream. This eliminates the mode search, and gives a good approximation of the quality of the FDR architecture. For MPEG-2, this was described in [1]. The architecture for a requantization transcoder with mode re-use (MR) in H.264/AVC is shown in Fig. 2, where $\Im_p(.)$ denotes the pixel-domain intra prediction operator.

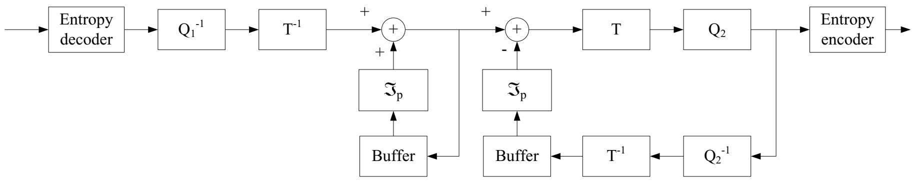

Fig. 2. Requantization transcoder with mode re-use

For low-delay applications, it is useful to further reduce the computational complexity of the transcoder. In this context, we here introduce a requantization transcoder with pixel-domain compensation. This architecture is shown in Fig. 3. When compared to the open-loop requantization transcoder, the PDC architecture restrains the drift of accumulated errors by compensating the dequantized 4×4 block of pixels in the pixel domain by a mode-dependent matrix ϕ. The computational advantage of this architecture over the MR transcoder is that intra prediction has to be performed only once. Additionally, in the MR

[1] After the inverse transform, the residual values are downscaled by 64.

transcoder, one buffer has to be maintained at the decoder side, and one at the encoder side. The PDC architecture halves the required memory, and the corresponding memory allocation and load and store operations.

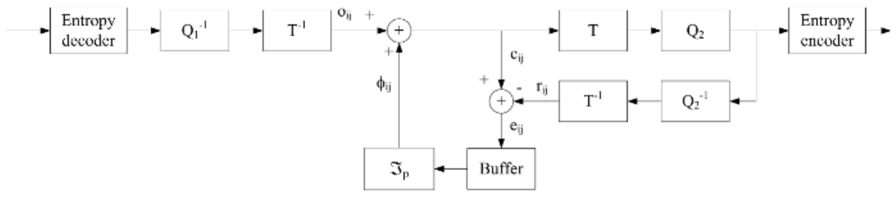

Fig. 3. Requantization transcoder with pixel-domain compensation

In order to obtain the compensation matrix ϕ, we first define the error values e_{ij} as the difference between the incoming residual information after inverse quantization, inverse transformation and drift compensation, and the corresponding requantized residual values after inverse quantization and inverse transformation, i.e., $e_{ij} = c_{ij} - r_{ij}$, for $i, j = 0, \ldots, 3$ (see Fig. 3). These quantization errors are stored in memory, and are used to compensate drift in the 4×4 blocks that depend on the current block.

The compensation matrix ϕ for a given 4×4 block is then constructed as follows. We select from the buffer the error values $e_{B,i,j}$, $e_{C,i,j}$, $e_{D,i,j}$, and $e_{E,i,j}$. By these, we denote the stored quantization errors e_{ij} of the 4×4 blocks B (top), C (upper-right), D (left), and E (upper-left), that surround the current 4×4 block A. For clarity, we only mention the error values that are required for the construction of the compensation, namely the error values $e_{B,3,j}$ and $e_{C,3,j}$ for $j = 0, \ldots, 3$, $e_{D,i,3}$ for $i = 0, \ldots, 3$, and $e_{E,3,3}$. These correspond to the positions that are normally used for intra 4×4 prediction. From these 13 values, the pixel-domain compensation matrix ϕ is constructed using the formulas for the intra 4×4 prediction, just as they are used in the encoder and decoder, but here applied on the smaller error values. For example, for horizontal prediction (mode 1), ϕ becomes:

$$\phi = \begin{bmatrix} e_{D,0,3} & e_{D,0,3} & e_{D,0,3} & e_{D,0,3} \\ e_{D,1,3} & e_{D,1,3} & e_{D,1,3} & e_{D,1,3} \\ e_{D,2,3} & e_{D,2,3} & e_{D,2,3} & e_{D,2,3} \\ e_{D,3,3} & e_{D,3,3} & e_{D,3,3} & e_{D,3,3} \end{bmatrix} .$$

In fact, ϕ is the difference of the intra prediction matrix of the original (unquantized) values, P, and the intra prediction matrix of the requantized values, P', i.e.,

$$\phi = \Im_p(\mathbf{e}) = \Im_p(\mathbf{c} - \mathbf{r}) = \Im_p(\mathbf{c}) - \Im_p(\mathbf{r}) = P - P' \quad ,$$

where $\mathbf{c}$ and $\mathbf{r}$ denote the compensated and requantized vectors of 13 prediction pixels used for prediction of the current 4×4 block. This equality holds exactly for prediction modes 0 and 1 (horizontal and vertical prediction). For modes 2 through 8, the right shift used for calculating the prediction may result in rounding errors.

3.3 Requantization Transcoder with Transform-Domain Compensation (TDC)

The transcoder with pixel-domain compensation as described in the previous section tries to overcome the quality-related problems of the open-loop requantization transcoder or the computational disadvantages of the MR transcoder. The question remains if it possible to further reduce the computational complexity of the PDC architecture. This reduction is possible by eliminating forward and inverse transforms, hence working as much as possible in the transform domain. This requantization transcoder with transform-domain compensation is visualized in Fig. 4.

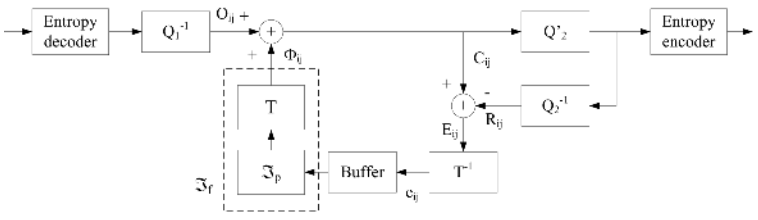

Fig. 4. Requantization transcoder with transform-domain compensation

This architecture is obtained, starting from the PDC architecture, through the following two steps. In the first step, forward and inverse transforms are moved to the feedback loop. This results in the elimination of one inverse transform. In the second step, we propose to combine the pixel-domain intra prediction $\Im_p$ and the forward transform T as indicated by the dashed line in Fig. 4. This results in transform-domain intra prediction $\Im_f$, which is for most of the prediction modes more efficient than the combination of the pixel-domain intra prediction $\Im_p$ and the forward transform T.

In order to achieve this, we need to obtain the transform-domain compensation matrix Φ, which can be obtained through the forward transform of the pixel-domain compensation matrix ϕ, i.e., $\Phi = C_f \phi C_f^T$. Because of the linear nature of the H.264/AVC integer transform, this matrix can also be written as the difference between the intra prediction matrices P and P' after transformation, i.e.,

$$\Phi = C_f \phi C_f^T = C_f (P - P') C_f^T = C_f P C_f^T - C_f P' C_f^T \quad .$$

Depending on the used prediction mode, the transformed coefficients will be compensated in a number of frequency-dependent positions. Using the above equation, in the case of the horizontal prediction (mode 1), we obtain:

$$\Phi = 4 \begin{bmatrix} e_{D,0,3} + e_{D,1,3} + e_{D,2,3} + e_{D,3,3} & 0 & 0 & 0 \\ 2e_{D,0,3} + e_{D,1,3} - e_{D,2,3} - 2e_{D,3,3} & 0 & 0 & 0 \\ e_{D,0,3} - e_{D,1,3} - e_{D,2,3} + e_{D,3,3} & 0 & 0 & 0 \\ e_{D,0,3} - 2e_{D,1,3} + 2e_{D,2,3} - e_{D,3,3} & 0 & 0 & 0 \end{bmatrix} .$$

For the vertical prediction (mode 0), Φ is constructed in a similar way. In the case of the DC prediction (mode 2), only the DC frequency position has to be compensated:

$$\Phi = 2 \begin{bmatrix} \sum_{j=0}^{3} e_{B,3,j} + \sum_{i=0}^{3} e_{D,i,3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} .$$

For the diagonal predictions (modes 3 and 4), Φ is a symmetric matrix, and thus can be calculated very efficiently. For the other predictions (modes 5 through 8), a larger number of multiplications is required. Depending on the cost of the multiplication on the target implementation platform, a trade-off between the transform-domain intra prediction $\Im_f$ and the combination of the pixel-domain intra prediction $\Im_p$ and the forward transform T can be made for these modes.

In order to calculate the prediction errors e_{ij}, one inverse integer transform for every 4×4 block is still performed. This is illustrated in Fig. 4. The inverse transform is kept to take full advantage of the properties of the H.264/AVC intra 4×4 prediction. We could, as in [9], have opted to perform the intra prediction on transformed pixels. This, however, would require storage of all 16 transformed coefficients E_{ij}. Because of the inverse transform, we only have to store 7 pixels for every 4×4 block. Apart for the lower memory requirements, it also has positive implications on the inverse transform, which now requires only 46 addition and 19 shift operations, instead of the complete inverse transform, which uses 80 addition and 32 shift operations. Another drawback of [9] is that the complete transform-domain intra prediction involves extensive floating-point multiplication, whereas the calculation in our architecture can be performed with integer arithmetic only.

The calculation of the compensation in the frequency domain should have no impact on the quality when compared to the PDC architecture. However, the frequency-domain operations are performed on inverse quantized values, which are upscaled by a factor 64 [6]. The non-linearity of the downscaling operation, which is performed after the inverse transform, introduces rounding errors, which could result in small quality differences between both architectures.

4 Experimental Results

In this section, we describe the results for the software implementation of our transcoding architectures for H.264/AVC bitstreams. The different transcoding architectures are tested using the video sequences Container and Stefan, both in CIF resolution. The objective quality and the bitrate of the transcoded bitstreams, using the PDC and TDC transcoding architectures, are compared with the results obtained through the FDR transcoder, the MR transcoder and the open-loop requantization transcoder.

The initial bitstreams were encoded using the JVT reference software (Joint Model 9.8), restricted to the intra 4×4 modes only. The bitstreams were then transcoded from the initial QP_1 to a higher QP_2 ($\Delta QP = QP_2 - QP_1$), using the five architectures. The rate-distortion performance for the transcoded Container video sequences is depicted in Fig. 5.

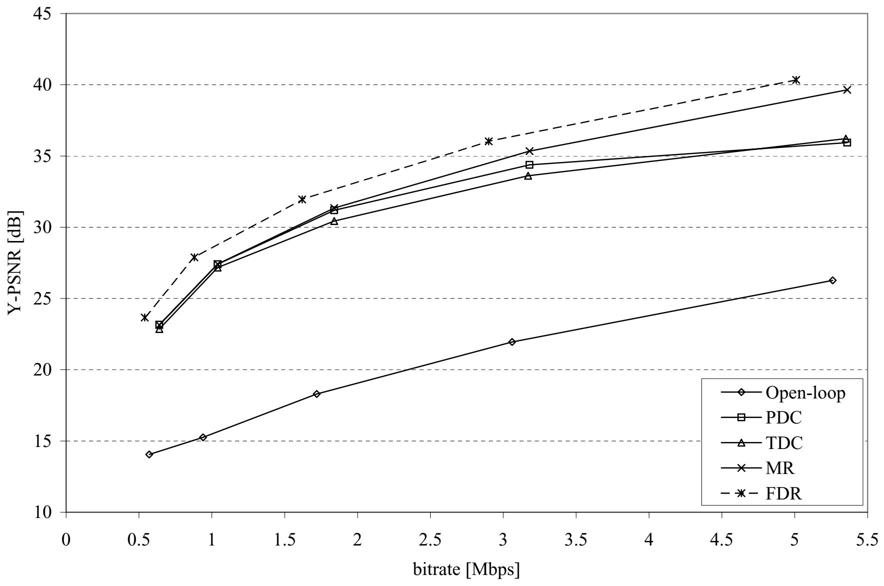

Fig. 5. Rate-distortion performance ($\Delta QP = 12$, Container sequence)

For completeness, the PSNR values (dB) and the bitrates (Mbps) for the different transcoding architectures using the video sequence Container are shown in Table 1 ($\Delta QP = 12$). The results obtained for the video sequence Stefan are similar, and are omitted here due to place constraints.

Table 1. PSNR [dB] and bitrate [Mbps] results (Container, $\Delta QP = 12$)

Original		FDR		MR		PDC		TDC		Open-loop	
QP_1	bitrate	PSNR	Bitrate	PSNR	Bitrate	PSNR	Bitrate	PSNR	Bitrate	PSNR	Bitrate
11	10.1	40.3	5.01	39.6	5.36	35.9	5.36	36.2	5.35	26.3	5.26
17	6.61	36.0	2.90	35.3	3.18	34.4	3.18	33.6	3.17	22.0	3.06
23	4.04	32.0	1.62	31.3	1.84	31.2	1.84	30.4	1.84	18.3	1.72
29	2.33	27.9	0.88	27.4	1.04	27.4	1.04	27.2	1.04	15.3	0.94
35	1.35	23.7	0.54	23.1	0.64	23.2	0.64	22.9	0.64	14.1	0.57

The FDR architecture generates qualitatively the best transcoded sequences over the full range of QPs. This architecture defines an upper bound for the objective quality of transcoded video sequences and will serve as reference in this section. For practical use, this type of transcoder is not feasible due to its high computational complexity, mainly originating from the exhaustive comparison of the prediction modes.

Although open-loop requantization transcoding of MPEG-2 bitstreams results in acceptable quality [1], the intra prediction in the H.264/AVC standard makes an open-loop architecture not suitable. The objective quality of the transcoded video sequences drops more than 10 dB in comparison with the objective quality obtained from the FDR transcoder. Hence, open-loop requantization transcoding of H.264/AVC intra coded pictures is no longer an option.

The positive effect of the drift compensation techniques used in the PDC and TDC requantization transcoders is clearly visible, when compared to the open-loop requantization architecture. For medium to high QPs (medium to low bitrates), the objective quality of the transcoded video sequences approximates the quality obtained through the FDR transcoder. The PSNR values vary little (less than 2 dB) from the quality obtained from the FDR transcoder. For lower QPs (high bitrates), the PSNR gap between the optimal FDR transcoder and the PDC and TDC architectures increases. This is caused by the non-linearity of the intra prediction formulas for modes 2 through 8, which become more dominant at higher bitrates.

In the case that complexity and memory requirements are not an issue, the MR transcoder can be used for high bitrate transcoding. The PSNR values of the MR transcoder differ approximately 1 dB from the PSNR values from the FDR architecture.

Table 1 also presents the bitrates originating from the different transcoders. The difference between the rate-distortion optimal FDR architecture and the four transcoder architectures is mainly caused by the suboptimal prediction modes for the target bitrate. The open-loop transcoder generates bitstreams with a bitrate which is approximately 5% larger than the bitstreams generated by the FDR architecture. The other three transcoding architectures, the MR transcoder and both the PDC and TDC transcoders generate bitrates which are between 5% and 10% larger than the optimal bitrates from the FDR architecture, due to the addition of new coefficients in the feedback-loop.

5 Conclusions

In this paper, requantization techniques for H.264/AVC bitstreams were discussed, focusing on the intra 4×4 prediction. Two architectures were presented that solve the problem of drift propagation, as encountered for the more traditional open-loop requantization transcoder. Results show that both architectures perform approximately equally well, and are able to approach the visual quality of a full decode and recode within 2 dB for medium to high QPs. Because of the reduced computational complexity of the proposed architectures over the FDR and MR architectures, they are highly suitable for on-the-fly rate reduction operations. As mentioned before, the TDC architecture is less complex than the PDC architecture, and therefore we suggest to use the former architecture.

Acknowledgements

The described research activities 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. Sun, H., Kwok, W., Zdepski, J.W.: Architectures for MPEG compressed bitstream scaling. IEEE Transactions on Circuits and Systems for Video Technology **6** (1996) 191–199
2. Vetro, A., Christopoulos, C., Sun, H.: Video transcoding architectures and techniques: an overview. IEEE Signal Processing Magazine (2003) 18–29
3. Nakajima, Y., Hori, H., Kanoh, T.: Rate conversion of MPEG coded video by requantization process. In: Proceedings of the 1995 International Conference on Image Processing, Washington, D.C. (1995)
4. Werner, O.: Requantization for transcoding of MPEG-2 intraframes. IEEE Transactions on Image Processing **8** (1999) 179–191
5. De Cock, J., Notebaert, S., Lambert, P., De Schrijver, D., Van de Walle, R.: 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) Accepted for publication
6. Malvar, H., Hallapuro, A., Karczewicz, M., Kerofsky, L.: Low-complexity transform and quantization in H.264/AVC. IEEE Transactions on Circuits and Systems for Video Technology **13** (2003) 598–603
7. Wiegand, T., Sullivan, G., Bjøntegaard, G., Luthra, A.: Overview of the H.264/AVC video coding standard. IEEE Transactions on Circuits and Systems for Video Technology **13** (2003) 560–576
8. Lefol, D., Bull, D., Canagarajah, N.: Performance evaluation of transcoding algorithms for H.264. IEEE Transactions on Consumer Electronics **52** (2006) 215–222
9. Chen, C., Wu, P.H., Chen, H.: Transform-domain intra prediction for H.264. In: Proceedings of the 2005 IEEE International Symposium on Circuits and Systems, Kobe, Japan (2005)